NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

# A ROBUST SYMMETRICAL NUMBER SYSTEM WITH GRAY CODE PROPERTIES FOR APPLICATIONS IN SIGNAL PROCESSING

by

Ilker Aydin Akin

September, 1997

Thesis Advisor: Phillip E. Pace

| REPORT DOCUMENTATION PAGE | Form Approved OMB No. 0704-0188 |
|---|---|

| | | |
|---|---|---|
| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE September 1997 | 3. REPORT TYPE AND DATES COVERED Master's Thesis |

| | |
|---|---|
| 4. TITLE AND SUBTITLE A ROBUST SYMMETRICAL NUMBER SYSTEM WITH GRAY CODE PROPERTIES FOR APPLICATIONS IN SIGNAL PROCESSING | 5. FUNDING NUMBERS |
| 6. AUTHOR(S) Akin, Ilker A. | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000 | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Research and Development Office | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER |

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

| | |
|---|---|
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited. | 12b. DISTRIBUTION CODE |

13. ABSTRACT *(maximum 200 words)*

A new symmetrical number system with applications in parallel signal processing is investigated. The "Robust Symmetrical Number System" (RSNS) is a modular system in which the integer values within each modulus, when considered together, change one at a time at the next position (Gray code properties). Although the observed dynamic range of the RSNS is somewhat less than the optimum symmetrical number system, the Gray code properties make it particularly attractive for folding analog-to-digital converters. With the RSNS, the encoding errors (due to comparator thresholds not being crossed simultaneously) are eliminated, as is the need for the corresponding interpolation signal processing (reduced complexity). Computer generated data is used to help determine the properties of the RSNS. These properties include the largest dynamic range (number of distinct consecutive vectors), and the position of the largest dynamic range within the system. The position of the maximum unambiguous dynamic range is also quantified. Least squares analysis of 2 and 3 moduli systems are used to formulate closed-form expressions for the dynamic range. To compare the advantages of the RSNS with previously published results, the transfer function of a 3-channel RSNS folding analog-to-digital converter architecture ($m_1 = 3$, $m_2 = 4$, and $m_3 = 5$) is numerically evaluated using SPICE.

| | |
|---|---|
| 14. SUBJECT TERMS Residue Number System, Robust Symmetrical Number System, Relatively Prime Numbers, Pairwise Relatively Prime Numbers, Moduli, Analog-to-Digital Converter, Dynamic Range, Point Dynamic Range, Fundamental Period, Point Index, Gray Code, Least Significant Bit | 15. NUMBER OF PAGES 102 |
| | 16. PRICE CODE |

| | | | |
|---|---|---|---|
| 17. SECURITY CLASSIFICATION OF REPORT Unclassified | 18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified | 19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified | 20. LIMITATION OF ABSTRACT UL |

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

# A ROBUST SYMMETRICAL NUMBER SYSTEM WITH GRAY CODE PROPERTIES FOR APPLICATIONS IN SIGNAL PROCESSING

Ilker Aydin Akin
Lieutenant Junior Grade, Turkish Navy
B.S., Turkish Naval Academy, 1991

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN SYSTEMS ENGINEERING**

from the

**NAVAL POSTGRADUATE SCHOOL**
**September 1997**

Author: [illegible]
Ilker Aydin Akin

Approved by: [illegible]
Phillip E. Pace, Thesis Advisor

David Styer
David Styer, Second Reader

[illegible]
Fred H. Levien, Chairman
Information Warfare Academic Group

## ABSTRACT

A new symmetrical number system with applications in parallel signal processing is investigated. The "Robust Symmetrical Number System" (RSNS) is a modular system in which the integer values within each modulus, when considered together, change one at a time at the next position (Gray code properties). Although the observed dynamic range of the RSNS is somewhat less than the optimum symmetrical number system, the Gray code properties make it particularly attractive for folding analog-to-digital converters. With the RSNS, the encoding errors (due to comparator thresholds not being crossed simultaneously) are eliminated, as is the need for the corresponding interpolation signal processing (reduced complexity). Computer generated data is used to help determine the properties of the RSNS. These properties include the largest dynamic range (number of distinct consecutive vectors), and the position of the largest dynamic range within the system. The position of the maximum unambiguous dynamic range is also quantified. Least squares analysis of 2 and 3 moduli systems is used to formulate closed-form expressions for the dynamic range. To compare the advantages of the RSNS with previously published results, the transfer function of a 3-channel RSNS folding analog-to-digital converter architecture ($m_1 = 3$, $m_2 = 4$, and $m_3 = 5$) is numerically evaluated using SPICE.

# TABLE OF CONTENTS

# ACKNOWLEDGMENTS

This thesis is dedicated to my parents whose guidance and encouragement inspire me to make them proud.

I would like to express my thanks to Professor Phillip E. Pace and Professor David Styer for their help and encouragement during this study. Their technical expertise and insights were invaluable to me. Their patience, maturity, and understanding of human nature helped me to overcome many difficulties.

This work is sponsored by the Research and Development Office.

# I. INTRODUCTION

## A. MODULAR NUMBER SYSTEMS

This study addresses a new modular number system. Before going into the details of the new system, it is useful to explain the properties of the some of the other modular number systems and examine their applications in Signal Processing. One of the drawbacks of the modular signal processing is the introduction of large errors (encoding errors) in the decoded binary or decimal values when only small errors occur within each modulus. The encoding errors and their elimination are the main reason for defining a new modular number system.

### 1. Residue Number System

The residue number system (RNS) is representative of a sawtooth folding waveform with folding period equal to the modulus *m*. An introduction to the RNS has been given by Szabo [1]. It has been established as an important tool in parallel processing applications and high-speed computations. The RNS can serve as a source for parallel operations (moduli) that are of smaller computational complexity. Each parallel operation has a different modulus and requires only a precision in accordance with that modulus. The folding waveform of the RNS is generated as

$$a \equiv r \,(mod\, m) \tag{1}$$

where $0 \leq r \leq m$, where $m$ is the modulus (channel value) which are PRP and $r$ is the residue of $a$ modulo $m$. Figure 1.1 shows the RNS folding waveforms for $m_1 = 4$ and $m_2 = 5$. Going from integer values within each modulus to decimal value requires the Chinese Remainder Theorem (CRT) [2-5]. The integers $r$ within each channel (modulus) are representative of a folded waveform with the period

$$P_{RNS} = m \tag{2}$$

The dynamic range (region of no ambiguities) of the RNS is

$$M_{RNS} = \prod_{i=1}^{N} m_i \tag{3}$$

and $m_i$ is the $i^{th}$ pairwise relatively prime (PRP) modulus (channel value), $1 \leq i \leq N$.

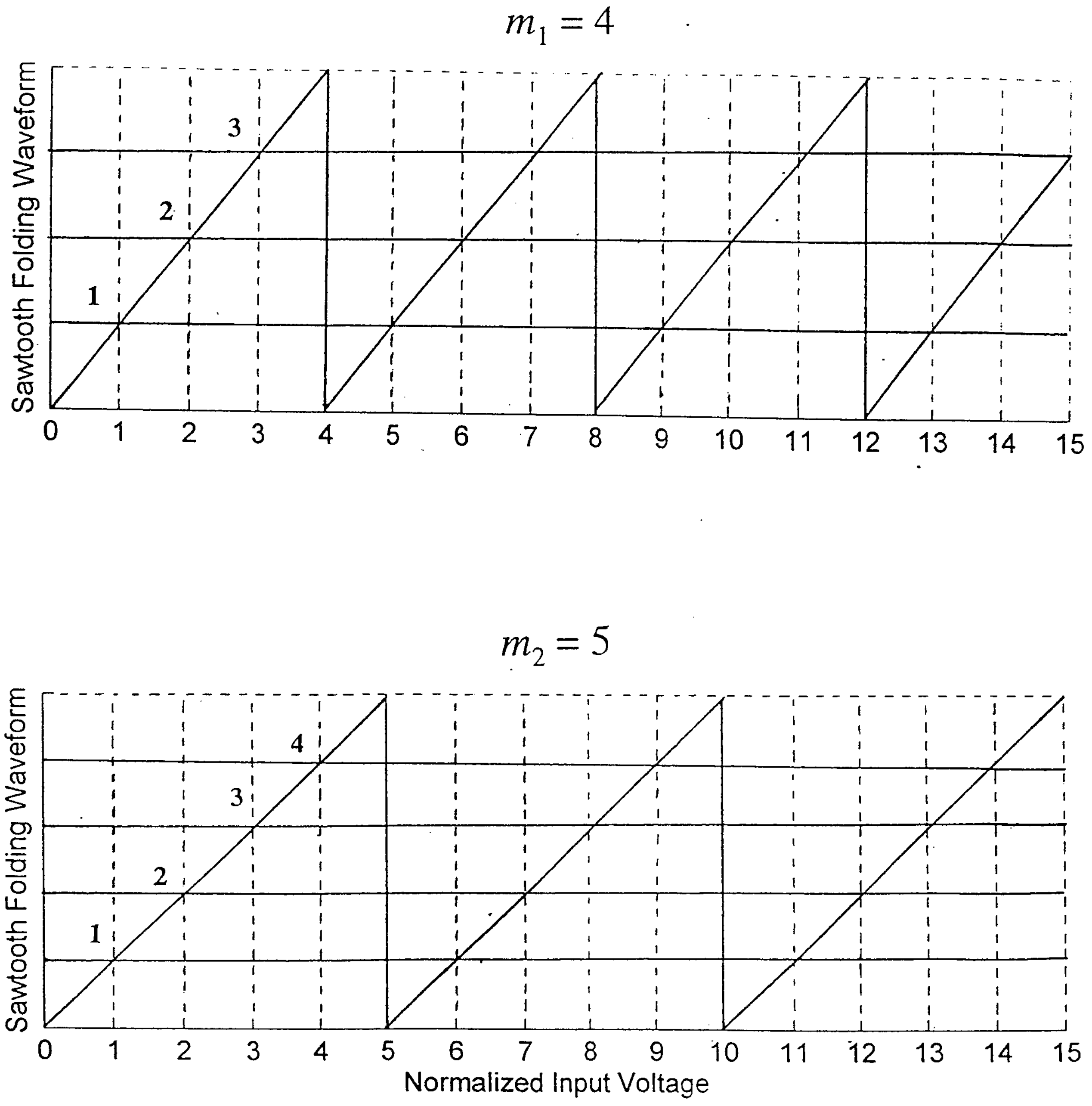


**Figure 1.1: The Folding Waveform of the RNS for** $m_1 = 4$ **and** $m_2 = 5$

## 2. Symmetrical Number System

The SNS is also composed of a number of pairwise relatively prime (PRP) moduli. The integers within each SNS modulus however are derived from a symmetrically folded waveform [6]. The integer values within each SNS modulus are derived from a symmetrical folding waveform and therefore incongruent modulus *m*. Due to the presence of ambiguities, the set of integers within each SNS modulus does not form a complete residue system by themselves. It is well known that the inclusion of additional redundant moduli can effectively detect and correct errors that may occur within a RNS representation of a number. The SNS formulation is based on a similar concept, which allow the ambiguities that arise within the SNS to be resolved by using various arrangements of the SNS moduli. The integer values within each SNS folding waveform is given as

$$x_{m,odd} = \left[0,1,\ldots,\left\lfloor\frac{m}{2}\right\rfloor,\left\lfloor\frac{m}{2}\right\rfloor,\ldots,2,1\right], \text{ when } m \text{ is odd} \tag{4}$$

$$x_{m,even} = \left[0,1,\ldots,\frac{m}{2},\frac{m}{2}-1,\ldots,2,1\right], \text{ when } m \text{ is even} \tag{5}$$

where $\lfloor x \rfloor$ indicates the greatest integer less than or equal to *x*. Both have size $1 \times m$ and consist of the symmetrical remainder elements $x_h$, $0 \leq h < m$. Figure 1.2 shows the SNS folding waveform for $m_1 = 4$ and $m_2 = 5$.

The integers within each modulus are representative of a symmetrically folded waveform with the period of the waveform equal to the modulus

$$P_{SNS} = m \tag{6}$$

where *m* is the modulus (channel).

The dynamic range of the SNS $\hat{M}_{SNS} < M_{OSNS}$ is given as

(i) *For PRP moduli $m_i$, with one of the moduli even, then*

$$\hat{M}_{SNS} = \min\left\{\frac{m_1}{2}\prod_{l=2}^{j} m_{i_l} + \prod_{l=j+1}^{N} m_{i_l}\right\} \tag{7}$$

*where j ranges from 1 to N -1 and $m_{i2}$, $m_{i3}$, ... $m_{iN}$ range over all permutations {2, 3, ..., N}. The product of j, from 2 to 1 is empty and its value is 1.*

(ii) *If all the PRP moduli in the system are odd, then*

$$\hat{M}_{SNS} = \min\left\{\frac{1}{2}\prod_{l=1}^{j} m_{i_l} + \frac{1}{2}\prod_{l=j+1}^{N} m_{i_l}\right\} \quad (8)$$

*where j ranges from 1 to N - 1 and $m_{i1}$, $m_{i2}$, $m_{i3}$, ... $m_{iN}$ range over all permutations { 1, 2, 3, ..., N}.*

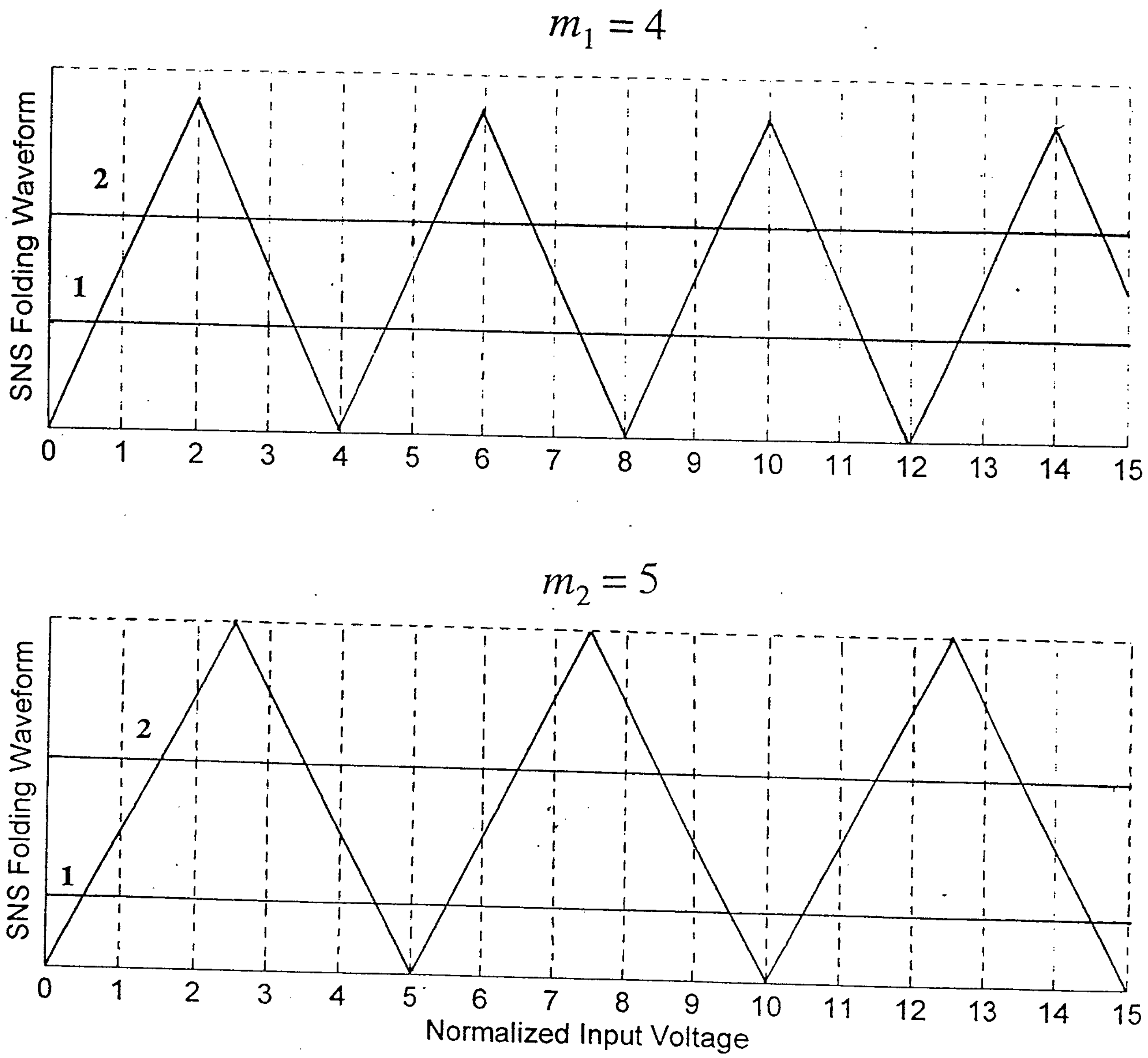


**Figure 1.2: The Folding Waveform of the SNS for $m_1 = 4$ and $m_2 = 5$**

## 3. Optimum Symmetrical Number System

The optimum symmetrical number system (OSNS) formulation is a direct consequence of the need to extract the maximum amount of information from a symmetrically folded waveform [7]. The OSNS scheme is also composed of a number of PRP moduli $m$. Integers within each OSNS modulus is defined as

$$x_m = [\, 0, 1, \ldots, m - 1, m - 1, \ldots, 1, 0]. \tag{9}$$

Figure 1.3 shows the OSNS folding waveform for $m_1 = 4$ and $m_2 = 5$. The integers within each SNS modulus are representative of a symmetrically folded waveform with the period of the waveform equal to the *twice* the modulus and

$$P_{OSNS} = 2m \tag{10}$$

where $m$ is the modulus. Due to the presence of ambiguities, the integers within (9) do not form a complete system of length $2m$ by themselves. The ambiguities that arise within the modulus are resolved by considering the paired values from all moduli together. By recombining the $N$ moduli, the OSNS is rendered a *complete* system having a one-to-one correspondence with the residue number system. For $N$ equal to the number of PRP moduli, the dynamic range of the OSNS is

$$M_{OSNS} = \prod_{i=1}^{N} m_i \tag{11}$$

where $m_i$ is the $i^{th}$ modulus, $1 \leq i \leq N$ and $N$ is the number of moduli. The dynamic range is also the position of the first repetitive moduli vector.

## 4. Application for Modular Number Systems in Signal Processing

A modular number system such as the RNS has been used in high performance digital signal processing [8]. Applications include a high rate arithmetic processor, digital filters, memory compression, and image processing. Recently, symmetrical number systems have been investigated in such application as Analog-to-Digital Converters (ADCs) [7, 9-13], Digital Antennas [14], Undersampling Receivers [6, 15], and Direction Finding Antennas [14]. The symmetrical number systems offer a considerable amount of flexibility in implementation. For higher resolution, faster conversion speed, and lower power dissipation the symmetrical number systems play an important role in folding ADCs applications.

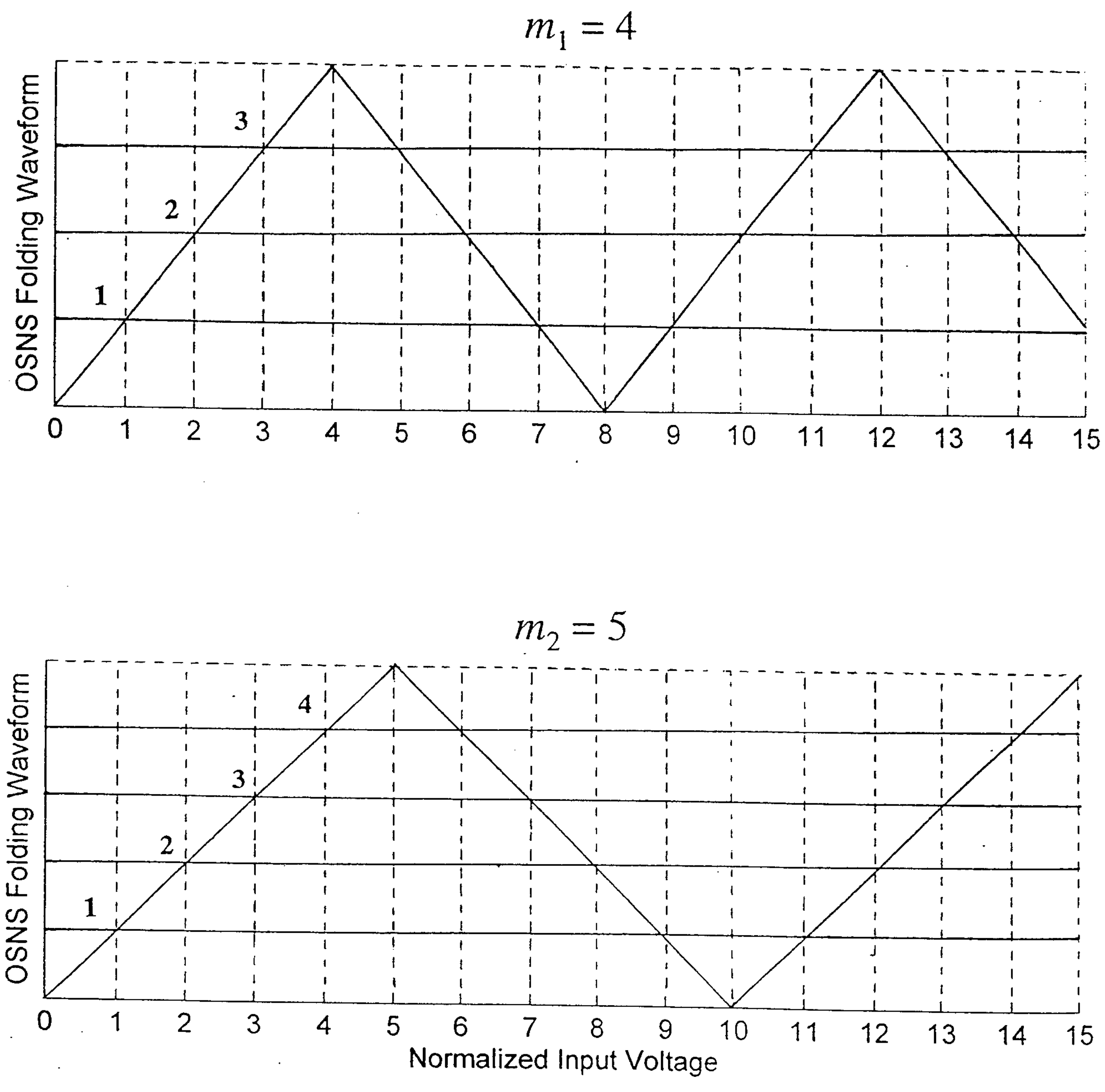


**Figure 1.3: The Folding Waveform of the OSNS for $m_1 = 4$ and $m_2 = 5$**

### 5. Encoding Errors of Modular Number Systems

Each of the above modular number systems for which the integer values within each modulus, when considered together, change one or more at the next position is shown in Table 1.1. Consequently, when some integers at a position to change, do change, while others do not, there will be a large error (encoding error). Table 1.1 shows the integer values within the modular systems for moduli $m_1 = 4$ and $m_2 = 5$.

When the input is 1, the integer values within the modular number systems are 1 for both moduli. By increasing the input from 1 to 2, all integer values within the modular number systems change from 1 to 2. If one of the integer values does not change, the new moduli pairs, for example (1,2) or (2,1) correspond to new input values. For the RNS, the input values can be 6 or 17. For the SNS, the input values can be 6, 13, 14, or 17. And for the OSNS, the input values can be 17 or 18. These values cause large errors (encoding errors) that must be dealt with. These large encoding errors present a number of difficulties when used for signal processing. The goal of this thesis is to develop a new number SNS formulation that minimizes the encoding errors. To examine the performance of the new number system formulation a folding ADC architecture is investigated.

## B. PRINCIPAL CONTRIBUTION

A new symmetrical number system with applications in parallel signal processing is investigated. The "Robust Symmetrical Number System" (RSNS) is a modular system in which the integer values within each modulus, when considered together, change one at a time at the next position (Gray code properties). Although the observed dynamic range of the RSNS is somewhat less than the optimum symmetrical number system, the Gray code properties make it particularly attractive for folding analog-to-digital converters. With the RSNS, the encoding errors (due to comparator thresholds not being crossed simultaneously) are eliminated, as is the need for the corresponding interpolation signal processing (reduced complexity). Computer generated data is used to help determine the properties of the RSNS. These properties include the largest dynamic range (number of distinct consecutive vectors), and the position of the largest dynamic range within the system. The position of the maximum unambiguous dynamic range is also quantified. Least squares analysis of 2 and 3 moduli systems is used to formulate closed-form expressions for the dynamic range. To compare the advantages of the RSNS with previously published results, the transfer function of a 3-channel RSNS folding analog-to-digital converter architecture ($m_1 = 3$, $m_2 = 4$, and $m_3 = 5$) is numerically evaluated using SPICE.

## C. THESIS OUTLINE

This thesis is organized in six parts. Chapter II provides the definition of the robust symmetrical number system (RSNS). An expression for the fundamental period is given and proven. In this chapter, the algorithm, which finds the dynamic range, point dynamic range, and fundamental period is discussed. The beginning and ending of the discrete states without redundancy for any given number of channel and shift values of the RSNS, is presented.

Chapter III investigates the 2-channel (moduli) RSNS. A specific number combination is considered. The RSNS performance using different moduli, the same distance apart is evaluated to find a closed-form expression dynamic range in the general case.

Chapter IV deals with the 3-channel (moduli) RSNS. A specific number combination is also considered in order to find a closed-form expression for the dynamic range.

Chapter V is based on the simulation of a 3-channel RSNS ADC in SPICE. The simulation channel and the shift values are $m_1 = 3$, $s_1 = 0$, $m_2 = 4$, $s_2 = 1$, and $m_3 = 5$, $s_3 = 2$, where $m$ shows the channel values and $s$ shows the shift. The folding circuit and comparator stages are designed to fold and quantize the incoming signal in a RSNS format.

Chapter VI summarizes the results and conclusions of this study and provides several recommendations for follow-up research.

| Input | RNS | | SNS | | QSNS | |
|---|---|---|---|---|---|---|
| | $m_1 = 4$ | $m_2 = 5$ | $m_1 = 4$ | $m_2 = 5$ | $m_1 = 4$ | $m_2 = 5$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 3 | 3 | 3 | 1 | 2 | 3 | 3 |
| 4 | 0 | 4 | 0 | 1 | 3 | 4 |
| 5 | 1 | 0 | 1 | 0 | 2 | 4 |
| 6 | 2 | 1 | 2 | 1 | 1 | 3 |
| 7 | 3 | 2 | 1 | 2 | 0 | 2 |
| 8 | 0 | 3 | 0 | 2 | 0 | 1 |
| 9 | 1 | 4 | 1 | 1 | 1 | 0 |
| 10 | 2 | 0 | 2 | 0 | 2 | 0 |
| 11 | 3 | 1 | 1 | 1 | 3 | 1 |
| 12 | 0 | 2 | 0 | 2 | 3 | 2 |
| 13 | 1 | 3 | 1 | 2 | 2 | 3 |
| 14 | 2 | 4 | 2 | 1 | 1 | 4 |
| 15 | 3 | 0 | 1 | 0 | 0 | 4 |
| 16 | 0 | 1 | 0 | 1 | 0 | 3 |
| 17 | 1 | 2 | 1 | 2 | 1 | 2 |
| 18 | 2 | 3 | 2 | 2 | 2 | 1 |
| 19 | 3 | 4 | 1 | 1 | 3 | 0 |
| 20 | 0 | 0 | 0 | 0 | 3 | 0 |
| 21 | 1 | 1 | 1 | 1 | 2 | 1 |
| 22 | 2 | 2 | 2 | 2 | 1 | 2 |

**Table 1.1: Modular Number Systems to Decimal Mapping for $m_1 = 4$ and $m_2 = 5$**

# II. ROBUST SYMMETRICAL NUMBER SYSTEMS

## A. DEFINING THE RSNS WAVEFORM.

In the robust symmetrical number system (RSNS), $N$ different periodic symmetrical waveforms are used with, pairwise relatively prime (PRP) integer, lengths $m_1, m_2, \ldots, m_N$. The RSNS is based on the following sequence

$$x_{m1} = [\,0, 1, 2, \ldots, m-1, m, m-1, \ldots, 2, 1] \qquad (12)$$

where $x_{m1}$ is the row vector and $m$ is a positive integer $(m > 0)$. In an $N$-channel RSNS, where $N \geq 2$, the basic sequence for $i^{th}$ channel, with modulus $m$ is

$$x_{m2}=[0, 0, \ldots, 0, 0, 1, 1, \ldots, 1, 1, \ldots, m, m, \ldots, m, m \ldots, 1, 1, \ldots, 1, 1]. \qquad (13)$$

In this sequence each value in the $x_{m1}$ row vector is put $N$ times in succession. This sequence is repeated in both directions, forming a periodic sequence with the period

$$P_{RSNS} = 2mN. \qquad (14)$$

An $N$-channel RSNS is formed of vectors by picking $N$ moduli (channels) $m_i$, and $N$ shift values $s_i$, $1 \leq i \leq N$, where $\{s_1, s_2, \ldots, s_N\}$ forms a complete set of residues modulo $N$. The $V_h$ vector

$$V_h = \begin{bmatrix} a_{1h} \\ \cdot \\ \cdot \\ \cdot \\ a_{Nh} \end{bmatrix} \qquad (15)$$

where $a_{ih}$ is the $(h - s_i)$ term in the periodic sequence for $m_i$, $1 \leq i \leq N$. Since the $i^{th}$ row is periodic with period $2Nm_i$, $1 \leq i \leq N$, it is clear that if the length that is simultaneously a multiple of $2Nm_1$, $2Nm_2, \ldots$, and $2Nm_N$ is taken, then the $N$-channel RSNS vector will repeat itself. Since the fundamental period for channel $i$ is $2Nm_i$, $1 \leq i \leq N$, it follows that a period for the RSNS vectors must be a multiple of each of $2Nm_i$. Therefore the *fundamental period* for the RSNS is

$$PF_{RSNS} = [2m_1N, 2m_2N, \ldots, 2m_NN] \qquad (16)$$

where $[n_1, n_2, \ldots, n_N]$ is the least common multiple of $n_1, n_2, \ldots, n_N$. From number theory [2-5],

$$PF_{RSNS} = 2N[m_1, m_2, \ldots, m_N]. \tag{17}$$

Considering a single channel the discrete states of the robust symmetrical number system are given as

$$g(m_i, s_i, n) = \begin{cases} \left\lfloor \dfrac{n - s_i}{N} \right\rfloor & \text{for } 1 + s_i \le n \le Nm_i + s_i \\ \left\lfloor \dfrac{2Nm_i + N - n + s_i - 1}{N} \right\rfloor & \text{for } Nm_i + 1 + s_i \le n \le 2Nm_i + s_i \end{cases} \tag{18}$$

where $m_i$ is the channel modulus and $s_i$ is the corresponding shift for

$$s_i \equiv 0, 1, 2, \ldots, N-1 \pmod N) \tag{19}$$

and $N \geq 2$ is the number of channels in the system. The values $\{s_1, s_2, \ldots, s_N\}$ must form a complete residue system modulo $N$. Here $n$ represents the normalized input voltage. Because of the relative property of the shifts, one of the shift values will be set equal to 0.

Upon inspection of any 2-channel and 3-channel RSNS, it is seen that if $V_h$ (the vector at position $h$) equals $V_{h+k}$, then $k$ is a multiple of 4 and 6, respectively. These are particular examples of a general rule:

***Rule 2.1:*** *Suppose that $V_h$, $V_{h+k}$ are the vectors at position $h$ and $h + k$, respectively, in an $N$-channel RSNS. If $V_h = V_{h+k}$, then $k$ is the multiple of $2N$.*

The proof is given below, but first recall that in an $N$-channel RSNS each term in a given channel occurs consecutively $N$ times. Consider the $N$ consecutive, equal terms form from left to right as lying in the first, second,..., $N^{th}$ position. Since $2N$ is a multiple of $N$, this rule shows that at any redundancy the redundant term lies in the same relative position as the original term. That is, if $V_h = V_{h+k}$,

$$V_h = \begin{bmatrix} a_{1h} \\ \cdot \\ \cdot \\ \cdot \\ a_{Nh} \end{bmatrix}, \; V_{h+k} = \begin{bmatrix} a_{1,h+k} \\ \cdot \\ \cdot \\ \cdot \\ a_{N,h+k} \end{bmatrix} \tag{20}$$

and $a_{A,h}$ occurs in the $i^{th}$ position in row $A$. Then, $a_{A,h} = a_{A,h+k}$ and $a_{A,h+k}$ also occurs in the $i^{th}$ position in row $A$, $1 \leq A \leq N$.

The proof of the *Rule 2.1*:

In the fundamental building block of the RSNS, the sequence by (12), the value at $j$ $(0 < j < m)$ equals to the value at $(j + k)$ when $j + k = m + (m - j)$, i.e. when $k = 2m - 2j$, an even number. This fundamental building block has length $2m$, and it is extended periodically. The values 0, and $m$ repeat every $2m$ terms, and no more. Other values repeat twice as often, but always an even number of terms apart. When the sequence is expanded by the $N$-fold repetition of each term, then all terms that repeat in the same relative position are a multiple of $2N$ apart.

Suppose that $a_h = a_{h+k}$, $k > 0$, and that $a_h$ is in the first (left) position. Then $a_{h+k}$ will be just $i$ terms to the right of the term in the first position, where $0 \leq i < N$. By the conclusion of the last paragraph $k = 2Nj + i$ for some integers $i$, $0 \leq i < N$, and $j \geq 0$ ( and not both 0). Similarly, if $a_h = a_{h+k}$, where $a_h$ is in the $N^{th}$ (right) position, then it follows that $k = 2Nj' - i'$, for some $i'$, $0 \leq i' < N$, and $j' > 0$.

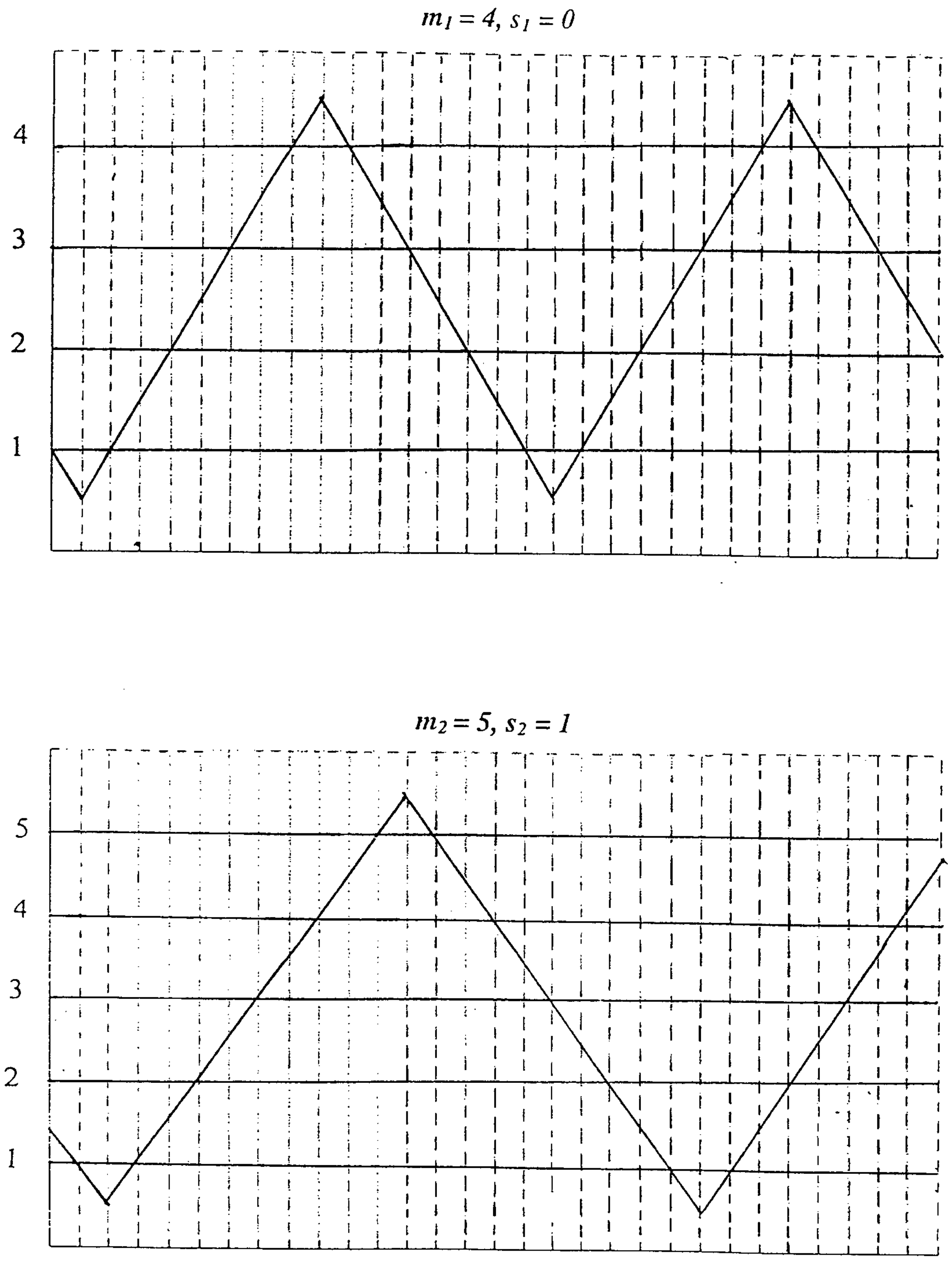
Now return to the full $N$-channel problem. Suppose $V_h = V_{h+k}$ with $k > 0$. By the definition of the RSNS one of the terms in (15) is in the first position, say $a_{A,h}$, and one is in the $N^{th}$ position, say $a_{B,h}$. Since $V_{h+k}$ equals $V_h$, $a_{A,h} = a_{A,h+k}$ and $a_{B,h} = a_{B,h+k}$. The first of these equations implies that $k = 2Nj + i$ for some $i$, $0 \leq i < N$ and $j \geq 0$, and the second equation implies that $k = 2Nj' - i'$, for some $i'$, $0 \leq i' < N$, and $j' > 0$. Putting these together: Since $2Nj + i$ and $2Nj' - i'$ are both equal to $k$, $2Nj + i = 2Nj' - i'$. Thus $i + i' = 2N(j' - j)$. However, $0 \leq i' + i < 2N$, so $i + i' = 0$ and $j = j'$. Since $i$ and $i'$ are nonnegative both must be zero. Therefore $k$ is a multiple of $2N$, as claimed.

The discrete states of the RSNS are indexed starting from the vector, which has all zero values. The index value starts from 0. These index values are put in a row matrix $h$, which lies in {..., -2, -1, 0, 1, 2, 3,...} and these values are called the ***point index***. For 2-channel RSNS, all point indexes determine 2 non-negative integer numbers, which are the discrete states of the RSNS. Similarly, $N$-channel RSNS point indexes determine $N$ discrete states of the RSNS.

Figure 2.1 shows the 2-channel RSNS folding waveforms and the discrete states within the RSNS. The thresholds shown on the vertical axis represent the integer value within each RSNS modulus. The integer values occur 2 times in succession. In this example the channel and corresponding shift values are $m_1 = 4$, $s_1 = 0$ and $m_2 = 5$, $s_2 = 1$.

Figure 2.2 shows 3-channel RSNS folding waveforms and the discrete states within the RSNS. The thresholds shown on the vertical axis represent the integer value within each RSNS modulus. The integer values occur 3 times in succession. In this example the channel and corresponding shift values are $m_1 = 3$, $s_1 = 0$, $m_2 = 4$, $s_2 = 1$ and $m_3 = 5$, $s_3 = 2$.

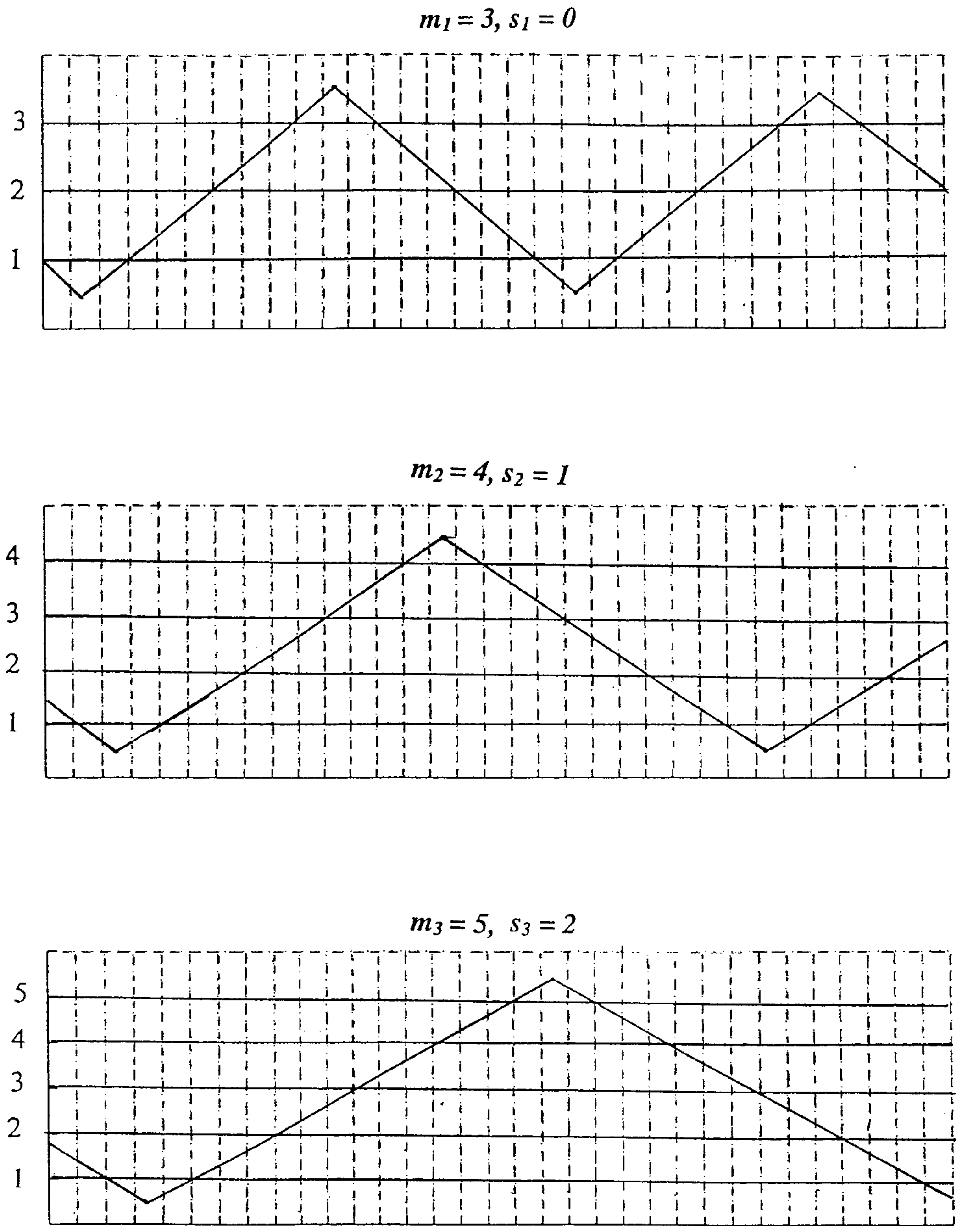
To demonstrate the performance of the 2 and 3-channel RSNS in detail, the RSNS computer algorithm is executed and Tables 2.1 and 2.2 show the distinct vector positions. Each row is considered separately. The first and second column of a given row give the beginning and ending point indexes containing consecutive sequences of distinct vectors of $N$ states. Furthermore, these positions can not be extended without introducing an ambiguity. The total length of the consecutive sequence of distinct vectors, starting at the beginning points (the point dynamic range) is shown in the third column. Note the maximal number of distinct vectors in Table 2.1 is 21 and in Table 2.2 is 43.



$m=4$ **0 0 1 1 2 2 3 3 4 4 3 3 2 2 1 1 0 0 1 1 2 2 3 3 4 4 3 3 2 2 1**$(s=0)$

$m=5$ **1 0 0 1 1 2 2 3 3 4 4 5 5 4 4 3 3 2 2 1 1 0 0 1 1 2 2 3 3 4 4**$(s=1)$

**Figure 2.1: The RSNS folding waveforms for** $m_1 = 4,\ s_1 = 0$ **and** $m_2 = 5,\ s_2 = 1$



| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *m=3* | **0** | **0** | **0** | **1** | **1** | **1** | **2** | **2** | **2** | **3** | **3** | **3** | **2** | **2** | **2** | **1** | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **2** | **2** | **2** | **3** | **3** | **3** | **2** | (*s=0*) |
| *m=4* | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **2** | **2** | **2** | **3** | **3** | **3** | **4** | **4** | **4** | **3** | **3** | **3** | **2** | **2** | **2** | **1** | **1** | **1** | **0** | **0** | **0** | **1** | **1** | (*s=1*) |
| *m=5* | **1** | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **2** | **2** | **2** | **3** | **3** | **3** | **4** | **4** | **4** | **5** | **5** | **5** | **4** | **4** | **4** | **3** | **3** | **3** | **2** | **2** | **2** | **1** | (*s=2*) |

**Figure 2.2: The RSNS folding waveforms for** $m_1 = 3$, $s_1 = 0$, $m_2 = 4$, $s_2 = 1$ **and** $m_3 = 5$, $s_3 = 2$

| Beginning Point Index | Ending Point Index | Total Length |
|---|---|---|
| 1 | 18 | 18 |
| 4 | 19 | 16 |
| 5 | 25 | 21 |
| 7 | 26 | 20 |
| 8 | 28 | 21 |
| 14 | 29 | 16 |
| 15 | 33 | 19 |
| 31 | 42 | 12 |

**Table 2.1: The Position of Distinct Vectors for the 2-Channel Case RSNS**

($m_1 = 4$, $s_1 = 0$, and $m_2 = 5$, $s_2 = 1$)

| Beginning Point Index | Ending Point Index | Total Length |
|---|---|---|
| 2 | 29 | 28 |
| 7 | 30 | 24 |
| 8 | 41 | 34 |
| 13 | 51 | 39 |
| 23 | 52 | 30 |
| 24 | 57 | 34 |
| 41 | 65 | 25 |
| 61 | 103 | 43 |

**Table 2.2: The Position of Distinct Vectors for the 3-Channel Case RSNS**

($m_1 = 3$, $s_1 = 0$, $m_2 = 4$, $s_2 = 1$, and $m_3 = 5$, $s_3 = 2$)

## B. THE DYNAMIC RANGE OF AN RSNS

The ***dynamic range*** of the RSNS is the maximum number of distinct vectors without a redundancy. The dynamic range of the RSNS is shown with the symbol $\hat{M}_{RSNS}$. The ***point dynamic range*** of the RSNS is the maximum number of consecutive vectors without a redundancy starting at a given vector (point index). The point dynamic range of the RSNS is shown with the symbol $\hat{M}_{P,RSNS}$.

The performance of a 2-channel RSNS is shown in Table 2.3 for $m_1 = 4$, $s_1 = 0$ and $m_2 = 5$, $s_2 = 1$. The selection of the shifts and their permutations among the 2-moduli has no effect on the dynamic range. The point index (beginning-ending) of the set of vectors that gives the dynamic range, however, is different.

| Moduli ($m_i$) | Shift ($s_i$) | Dynamic Range ($\hat{M}_{RSNS}$) | Fundamental Period ($P_{RSNS}$) | Beginning – Ending Point Index of Dynamic Range |
|---|---|---|---|---|
| 4<br>5 | 0<br>1 | 21 | 80 | 5-25 |
| 4<br>5 | 1<br>0 | 21 | 80 | 14-34 |

**Table 2.3: The Performance with the Permutation of Shifts for the 2-Channel Case RSNS**

The performance of a 3-channel RSNS is shown in Table 2.4 for $m_1 = 3$, $s_1 = 0$, $m_2 = 4$, $s_2 = 1$ and $m_3 = 5$, $s_3 = 2$. The selection of the shifts and their permutations among the 3-moduli also has no effect on the dynamic range. The point index (beginning-ending) of the set of vectors that determines the dynamic range is different.

| Moduli ($m_i$) | Shift ($s_i$) | Dynamic Range ($\hat{M}_{RSNS}$) | Fundamental Period ($P_{RSNS}$) | Beginning – Ending Point Index of Dynamic Range |
|---|---|---|---|---|
| 3<br>4<br>5 | 0<br>1<br>2 | 43 | 360 | 61-103 |
| 3<br>4<br>5 | 0<br>2<br>1 | 43 | 360 | 142-184 |
| 3<br>4<br>5 | 1<br>0<br>2 | 43 | 360 | 8-50 |
| 3<br>4<br>5 | 1<br>2<br>0 | 43 | 360 | 134-176 |
| 3<br>4<br>5 | 2<br>0<br>1 | 43 | 360 | 180-222 |
| 3<br>4<br>5 | 2<br>1<br>0 | 43 | 360 | 81-123 |

**Table 2.4: The Performance with the Permutation of Shifts for the 3-Channel Case RSNS**

## C. A WEAKNESS OF THE RSNS WITH RESPECT TO THE INCREASE OF THE NUMBER OF CHANNELS

The integers fall into two classes, even and odd parities. The integer values for any number of RSNS channels examined by a parity point of view. For the 2-channel ideal case, their parities distribute $2^2 = 4$ different ways. These are even-even, even-odd, odd-even, and odd-odd. The possible parity sets for the 2-channel RSNS are shown in Table 2.5 and these parity sets repeat themselves. In the 2-channel RSNS, the efficiency of the parity sets is same as the ideal one.

| Index | $m_1$ | $m_2$ |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 0 | 1 |
| 3 | 1 | 1 |
| 4 | 1 | 2 |
| 5 | 2 | 2 |
| 6 | 2 | 3 |
| 7 | 3 | 3 |
| 8 | 3 | 4 |
| 9 | 4 | 4 |
| 10 | 4 | 5 |
| 11 | . | . |
| 12 | . | . |
| 13 | . | . |

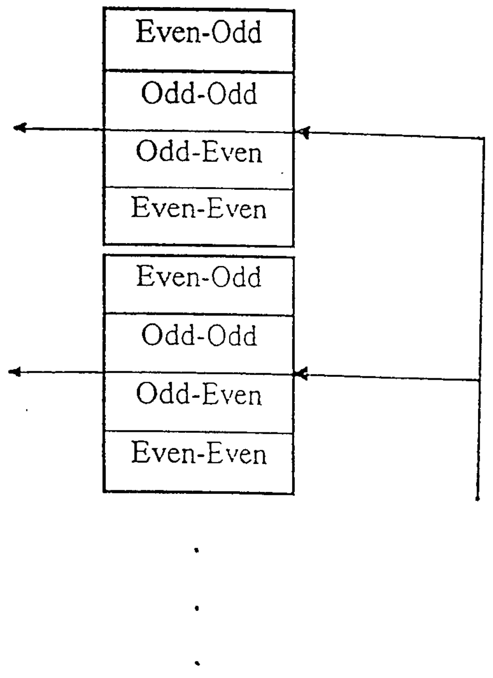


**Table 2.5: Demonstration of the parity combinations for $N = 2$ with $s_1 = 0$ and $s_2 = 1$**

For the 3-channel ideal case, their parities distribute $2^3 = 8$ different ways. These are even-even-even, even-even-odd, even-odd-odd, even-odd-even, odd-even-even, odd-even-odd, odd-odd-even, and odd-odd-odd. There are only 6 possible parity sets for the

3-channel RSNS (restricted parity combinations) and they repeat themselves as shown in Table 2.6. The parity sets even-odd-even and odd-even-odd will never show up in this 3-channel RSNS. In the 3- channel RSNS, the efficiency of the parity sets is 25% less than the ideal one.

| Index | $m_1$ | $m_2$ | $m_3$ |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 2 | 0 | 0 | 1 |
| 3 | 0 | 1 | 1 |
| 4 | 1 | 1 | 1 |
| 5 | 1 | 1 | 2 |
| 6 | 1 | 2 | 2 |
| 7 | 2 | 2 | 2 |
| 8 | 2 | 2 | 3 |
| 9 | 2 | 3 | 3 |
| 10 | 3 | 3 | 3 |
| 11 | 3 | 3 | 4 |
| 12 | 3 | 4 | 4 |
| 13 | . | . | . |
| 14 | . | . | . |
| 15 | . | . | . |

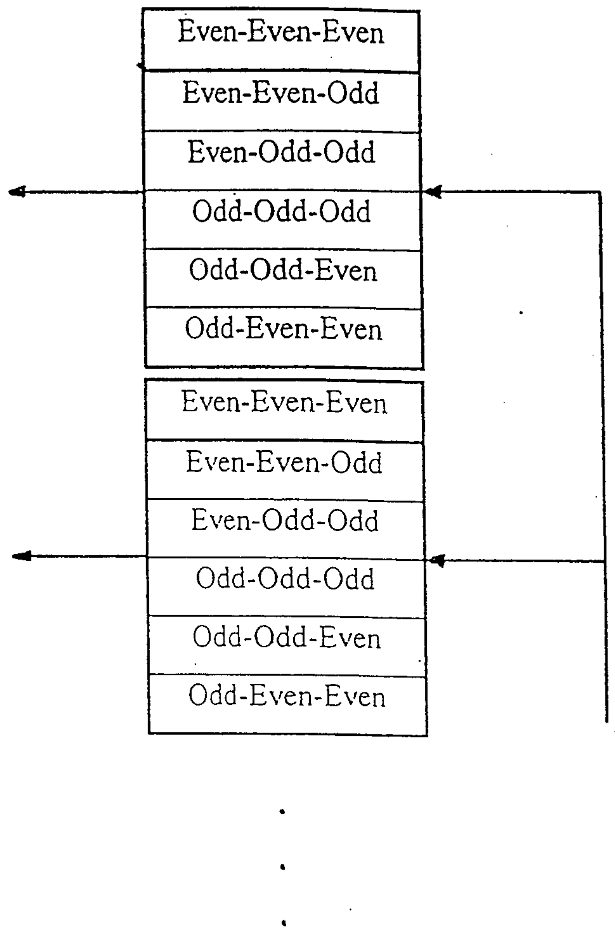


**Table 2.6: Demonstration of restricted parity combinations for $N = 3$ with $s_1 = 0$, $s_2 = 1$ and $s_3 = 2$**

In the $N$-channel RSNS each term in the vector $V_{h+N}$ by (20) differs by exactly one from corresponding term in $V_h$ by (15). Therefore, each term in $V_{h+N}$ has opposite parity from the corresponding term in $V_h$. It follows quickly that each term in $V_{h+2N}$ has exactly the same parity as the corresponding term in $V_h$. From this result, it can be said that the parities of the terms in any $V_h$ will be exactly the same as the parities of the corresponding terms in one of $V_1$, $V_2$, ..., $V_N$. This conclusion ends the proof that there will be just $2N$ different parity sets represented, out of a possible $2^N$ for $N$-channel RSNS.

## D. RSNS COMPUTER ALGORITHM

In this study, one of the important goals is to be able to find a closed-form expression for the dynamic range and fundamental period of any number of RSNS channels. The fundamental period of any number of RSNS is given in Section A. The RSNS computer algorithm (**rsns.m**) is constructed in order to examine the performance of the RSNS and find the dynamic range for any $N \geq 2$ RSNS systems.

In **rsns.m**, the channel and shift values, which are going to be used must be entered and the discrete states of the RSNS are calculated by using these values along with (18). The calculated discrete states of the RSNS are put into a matrix. This matrix is extended two times the *fundamental period* of the RSNS channels.

The next step in the algorithm is to find the point index of the discrete states within the extended matrix. After finding the point index, the data is put into a new matrix. The point index of each discrete state and its redundancy is assigned as the beginning and ending point index, respectively.

The values of the new matrix are sorted and rearranged such, that the beginning and ending point index are monotonically increasing. Then, the length at each point index is found (Ending point index – Beginning point index + 1). The maximum value of these lengths is the dynamic range of the RSNS.

The algorithm **rsns.m** also displays the dynamic range, fundamental period, entered channel and shift values, the beginning-ending point index values of the discrete states without redundancy and the point dynamic range of the RSNS. The program structure and listing of the **rsns.m** is shown in Appendix. Tables 2.1 and 2.2 are the example runs of the **rsns.m** algorithm.

# III. 2-CHANNEL ROBUST SYMMETRICAL NUMBER SYSTEM

## A. SPECIAL CASE: MODULI SPACING 1 OR 2 APART

In this chapter, the 2-channel RSNS is addressed. The 2 moduli are RP numbers and the different pairs are chosen to have the same distance between each other. The algorithm **rsns.m** is used to find the dynamic range. When the moduli have a difference ($\Delta$) of 1 or 2, the following computer results have been observed as shown in Table 3.1 and 3.2.

| $m_1$ | $m_2$ | Dynamic Range | 1st Difference |
|---|---|---|---|
| 3 | 4 | 15 | 6 |
| 4 | 5 | 21 | 6 |
| 5 | 6 | 27 | 6 |
| 6 | 7 | 33 | 6 |
| $\hat{M}_{RSNS} = 3m_1 + 3m_2 - 6$ | | | |

**Table 3.1: The dynamic range of $m_1$ and $m_2 = m_1 + 1$ where $m_1 \geq 3$ ($\Delta = 1$)**

| $m_2$ | $m_1$ | Dynamic Range | 1st Difference |
|---|---|---|---|
| 7 | 5 | 29 | 12 |
| 9 | 7 | 41 | 12 |
| 11 | 9 | 53 | 12 |
| 13 | 11 | 65 | 12 |
| $\hat{M}_{RSNS} = 3m_1 + 3m_2 - 7$ | | | |

**Table 3.2: The dynamic range of $m_1$ and $m_2 = m_1 + 2$ where $m_1 \geq 5$ ($\Delta = 2$)**

These relationships show a uniform increase of 3 with each unit increase of value in $m_1$ and $m_2$. The formulas

$$\hat{M}_{RSNS} = 3m_1 + 3m_2 - 6 \tag{21}$$

$$\hat{M}_{RSNS} = 3m_1 + 3m_2 - 7 \tag{22}$$

will work for the first and second sets of cases which are shown in Table 3.1 and 3.2, respectively. These cases can be expressed in terms of the lower value, $m_1$ and a difference, $\Delta = m_2 - m_1$. In the first set $\Delta = 1$, and in the second set $\Delta = 2$. Thus, (21) and (22) can be written as

$$\hat{M}_{RSNS} = 3m_1 + 3(m_1 + \Delta) - 6 = 6m_1 - 3 \tag{23}$$

$$\hat{M}_{RSNS} = 3m_1 + 3(m_1 + \Delta) - 7 = 6m_1 - 1. \tag{24}$$

These can be combined as

$$\hat{M}_{RSNS} = 6m_1 + 2\Delta - 5 \tag{25}$$

and now $\Delta$ can be replaced with $m_2 - m_1$, yielding

$$\hat{M}_{RSNS} = 6m_1 + 2(m_2 - m_1) - 5 = 4m_1 + 2m_2 - 5 \tag{26}$$

and this formula works for both $\Delta = 1$ and $\Delta = 2$. This formula is checked against the computer generated results. For example, when $m_1=100$ and $m_2=101$, the dynamic range is 597 which fits (26).

## B. SPECIAL CASE: MODULI SPACING 3 OR MORE APART

The following results have been observed, when the channel numbers are 3, 4 and 5 distance apart as shown in Table 3.3, 3.4 and 3.5.

| $m_1$ | $m_2$ | Dynamic Range | 1st Difference |
|---|---|---|---|
| 5 | 8 | 34 | 18 |
| 8 | 11 | 52 | 18 |
| 11 | 14 | 70 | 18 |
| 14 | 17 | 88 | 18 |
| $\hat{M}_{RSNS} = 3m_1 + 3m_2 - 5$ | | | |

**Table 3.3: The dynamic range of $m_1$ and $m_2 = m_1 + 3$ where $m_1 \geq 5$ $(\Delta = 3)$**

| $m_1$ | $m_2$ | Dynamic Range | 1st Difference |
|---|---|---|---|
| 7 | 11 | 48 | 24 |
| 11 | 15 | 72 | 24 |
| 15 | 19 | 96 | 24 |
| 19 | 23 | 120 | 24 |
| $\hat{M}_{RSNS} = 3m_1 + 3m_2 - 6$ | | | |

**Table 3.4: The dynamic range of $m_1$ and $m_2 = m_1 + 4$ where $m_1 \geq 7$ $(\Delta = 4)$**

| $m_1$ | $m_2$ | Dynamic Range | 1st Difference |
|---|---|---|---|
| 7 | 12 | 50 | 30 |
| 12 | 17 | 80 | 30 |
| 17 | 22 | 110 | 30 |
| 22 | 27 | 140 | 30 |
| $\hat{M}_{RSNS} = 3m_1 + 3m_2 - 7$ | | | |

**Table 3.5: The dynamic range of $m_1$ and $m_2 = m_1 + 5$ where $m_1 \geq 7$ $(\Delta = 5)$**

These relationships show a uniform increase of 3 with each unit increase of value in $m_1$ and $m_2$. The formulas

$$\hat{M}_{RSNS} = 3m_1 + 3m_2 - 5 \tag{27}$$

$$\hat{M}_{RSNS} = 3m_1 + 3m_2 - 6 \tag{28}$$

$$\hat{M}_{RSNS} = 3m_1 + 3m_2 - 7 \tag{29}$$

will work for the first, second and third sets of cases which are shown in Table 3.3, 3.4 and 3.5, respectively..

As in the earlier example these cases can be expressed in terms of the lower value, $m_1$ and a difference, $\Delta = m_2 - m_1$. In the first set $\Delta = 3$, in the second set $\Delta = 4$, and in the third set $\Delta = 5$. Thus, (27), (28) and (29) can be written as

$$\hat{M}_{RSNS} = 3m_1 + 3(m_1 + \Delta) - 5 = 6m_1 + 4 \quad (30)$$

$$\hat{M}_{RSNS} = 3m_1 + 3(m_1 + \Delta) - 6 = 6m_1 + 6 \quad (31)$$

$$\hat{M}_{RSNS} = 3m_1 + 3(m_1 + \Delta) - 7 = 6m_1 + 8. \quad (32)$$

These can be combined as

$$\hat{M}_{RSNS} = 6m_1 + 2\Delta - 2 \quad (33)$$

and now $\Delta$ can be replaced with $m_2 - m_1$, yielding

$$\hat{M}_{RSNS} = 6m_1 + 2(m_2 - m_1) - 2 = 4m_1 + 2m_2 - 2. \quad (34)$$

and this formula works for all three values of $\Delta$, $\Delta = 3$, $\Delta = 4$, and $\Delta = 5$. This formula is checked against the computer generated results. For example, when $m_1=101$ and $m_2=105$, the dynamic range is 612 which fits (34). It is conjectured that (34) is valid for any $\Delta \geq 3$. Table 3.6 shows a summary of the specific 2-channel dynamic range formulas.

| Spacing between moduli ($\Delta$ ) | Dynamic Range Formula |
|---|---|
| $\Delta = 1$ or $\Delta = 2$ | $\hat{M}_{RSNS} = 4m_1 + 2m_2 - 5$ |
| $\Delta \geq 3$ | $\hat{M}_{RSNS} = 4m_1 + 2m_2 - 2$ |

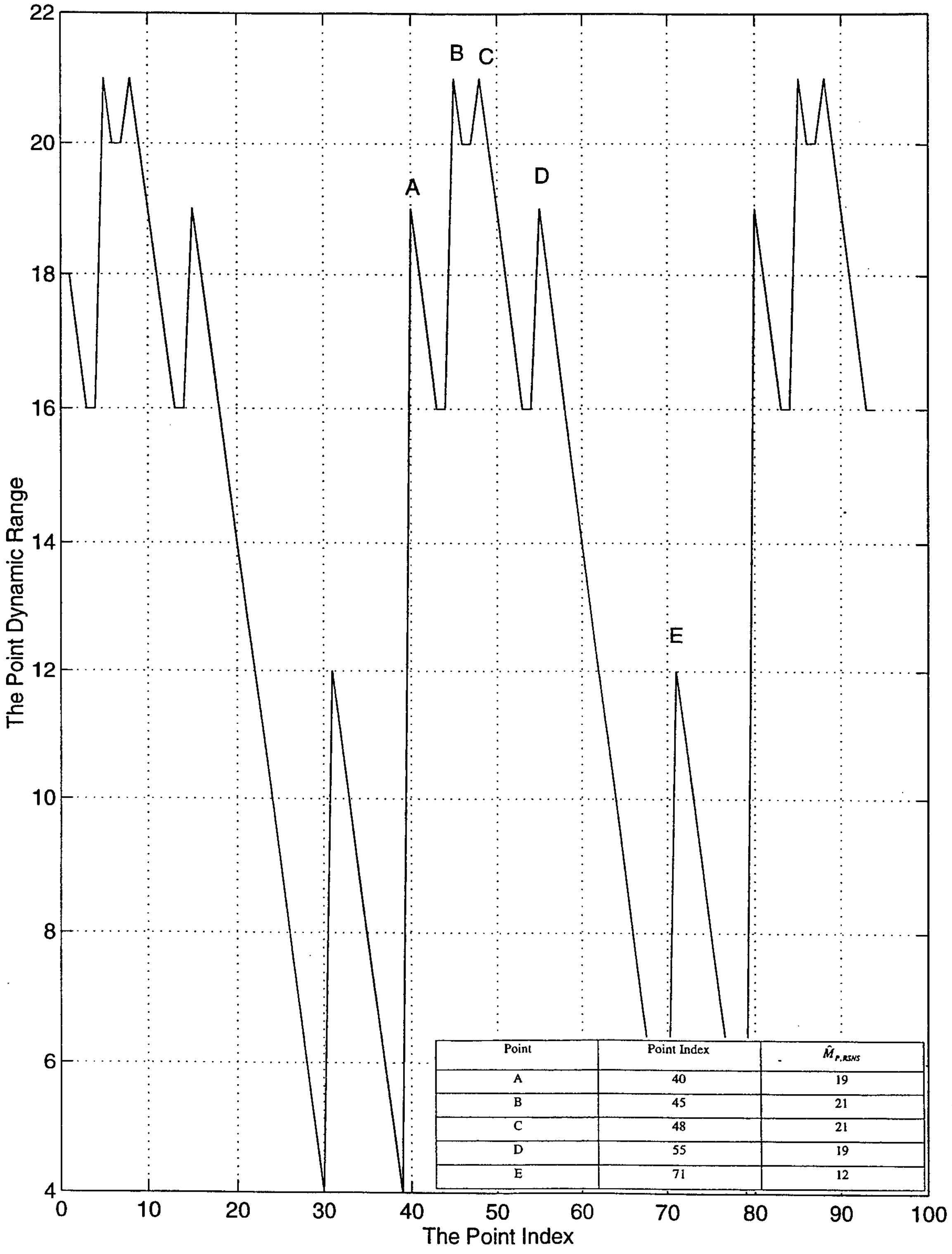
**Table 3.6: The Summary of the Dynamic Range Formulas for specific 2-channel RSNS**

## C. GRAPHICAL ANALYSIS

The algorithm **rsns.m** also finds and displays the *point dynamic range* of the 2-channel RSNS as explained in Chapter II. The point dynamic range versus the point index for the channel and shift values $m_1 = 4$, $s_1 = 0$ and $m_2 = 5$, $s_2 = 1$ are shown in Figures 3.1 and 3.2, respectively. The waveform with the points A, B, C, D and E, that are shown in Figures 3.1 and 3.2, construct a special waveform that repeats itself $\frac{P_{RSNS}}{2}$. This special waveform, the point dynamic range (PDR) waveform, gives the dynamic range starting at each point with increasing point index. The fundamental period of this particular 2- channel RSNS example is $PF_{RSNS} = 80$ by (16). So, this PDR waveform repeats itself every 40 points.

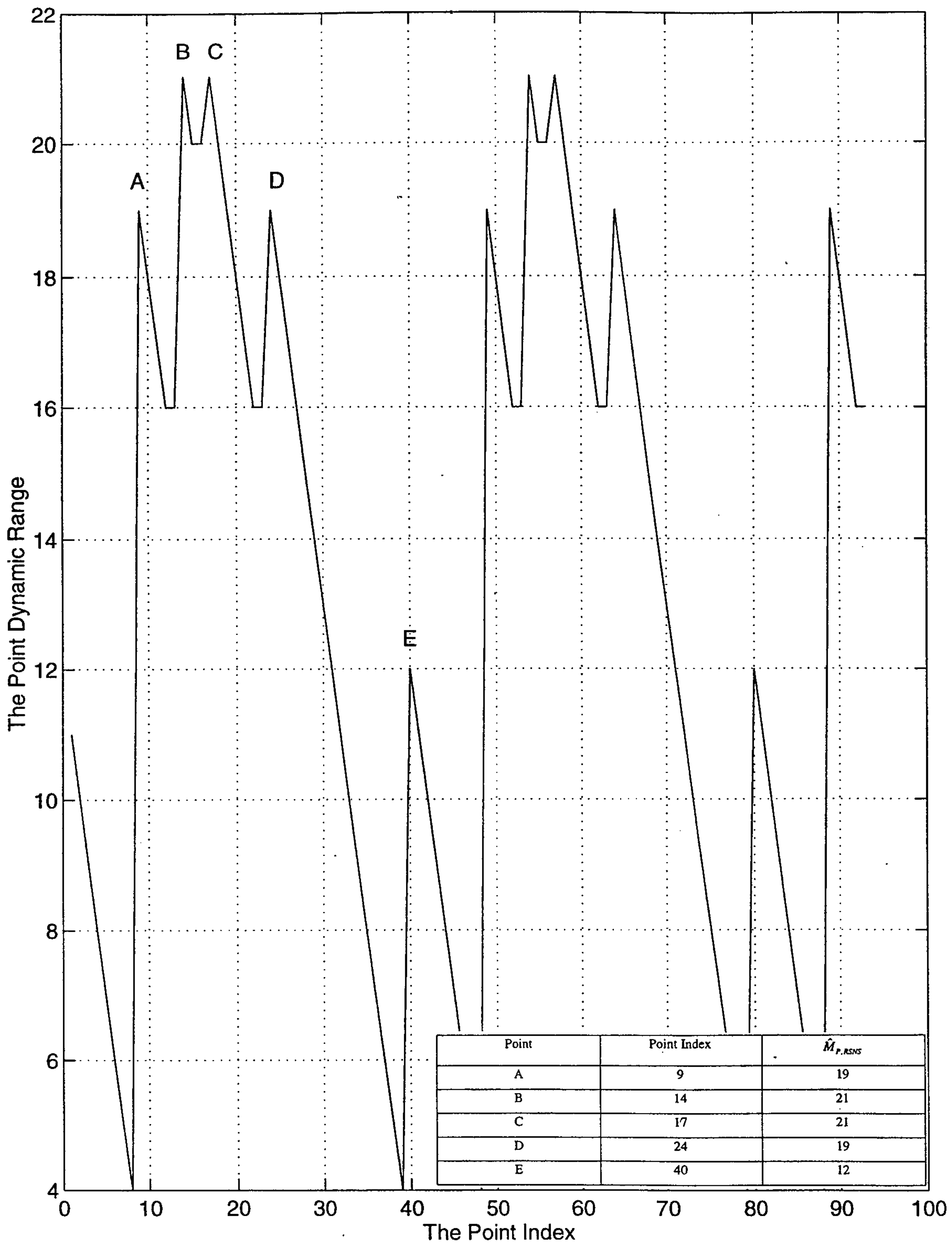
The maximum value of the point dynamic range is the dynamic range of the 2-channel RSNS. In this example, $\hat{M}_{RSNS} = 21$. The dynamic range point indexes are 5, 8, 45, 48, ... for $s_1 = 0$, $s_2 = 1$ and 14, 17, 54, 57, ... for $s_1 = 1$, $s_2 = 0$ which can be seen from Figure 3.1 and 3.2. These values are the beginning point index of the dynamic range. The dynamic range point index values are also half of the fundamental period apart from each other.

The PDR waveform of the point dynamic range of the 2-channel RSNS will be the same for different shift values as shown in Figure 3.1 and 3.2. The only difference is the point index (beginning value) of the dynamic range.



| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 40 | 19 |
| B | 45 | 21 |
| C | 48 | 21 |
| D | 55 | 19 |
| E | 71 | 12 |

**Figure 3.1: The Point Dynamic Range Waveform for** $m_1 = 4$, $s_1 = 0$,**and** $m_2 = 5$, $s_2 = 1$.



| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 9 | 19 |
| B | 14 | 21 |
| C | 17 | 21 |
| D | 24 | 19 |
| E | 40 | 12 |

**Figure 3.2: The Point Dynamic Range Waveform for** $m_1 = 4$, $s_1 = 1$,**and** $m_2 = 5$, $s_2 = 0$.

# IV. 3-CHANNEL ROBUST SYMMETRICAL NUMBER SYSTEM

## A. SPECIAL CASE: MODULI SPACING 1 APART

In this chapter, the 3-channel RSNS is addressed. An equal distance separates the moduli. When the PRP channel numbers are 1 apart and the set of the 3-channel RSNS are increasing uniformly, the following results have been observed as shown in Table 4.1.

| $m_1$ | $m_2$ | $m_3$ | Dynamic Range | 1st Difference | 2nd Difference |
|---|---|---|---|---|---|
| 3 | 4 | 5 | 43 | 39 | 12 |
| 5 | 6 | 7 | 82 | 51 | 12 |
| 7 | 8 | 9 | 133 | 63 | |
| 9 | 10 | 11 | 196 | | |
| $\hat{M}_{RSNS} = \frac{3}{2}m_1^2 + \frac{15}{2}m_1 + 7$ | | | | | |

**Table 4.1: The Dynamic Range of** $m_1$, $m_2 = m_1 + 1$, **and** $m_3 = m_1 + 2$ **where** $m_1 \geq 3$

The first difference (e.g., 82 – 43 = 39) and second difference (e.g., 51 – 39 = 12) of the dynamic range of the 3-channel RSNS for $m_1$, $m_2 = m_1 + 1$, and $m_3 = m_1 + 2$ where $m_1 \geq 3$ are shown in Table 4.1. The relationship between the dynamic range differences can be expressed as a 2nd degree polynomial

$$p(n) = an^2 + bn + c \tag{35}$$

where $n$ is the lowest channel value and $a$, $b$, and $c$ are a constant real numbers. Each set of 3-moduli RSNS values shows a uniform increase of 2. Therefore, if the first set of channel values corresponds to (35), then the second set of channel values corresponds to

$$\begin{aligned} p(n+2) &= a(n+2)^2 + b(n+2) + c \qquad (36) \\ &= a(n^2 + 4n + 4) + bn + 2b + c \\ &= an^2 + 4an + 4a + bn + 2b + c \end{aligned}$$

$$= p(n) + 4an + 4a + 2b.$$

The first difference of the sets ((36) – (35)) is

$$q(n) = p(n + 2) - p(n) = 4an + 4a + 2b. \tag{37}$$

If the first difference of the first two sets is (37), the first set difference of the second two sets can be written as

$$\begin{aligned} q(n + 2) &= 4a(n + 2) + 4a + 2b \\ &= 4an + 8a + 4a + 2b \\ &= q(n) + 8a. \end{aligned} \tag{38}$$

The second difference of the sets are unique and

$$q(n+2) - q(n) = 8a. \tag{39}$$

The constant "*a*" can be found by using (39)

$$\begin{aligned} q(n+2) - q(n) &= 8a = 12 \\ a &= \frac{12}{8} = \frac{3}{2} \end{aligned} \tag{40}$$

The constant "*b*" can be found by using (37) for $n = 3$ ( $m_1 = 3$)

$$\begin{aligned} p(n+2) - p(n) = 39 &= 4\left(\frac{3}{2}\right)3 + 4\left(\frac{3}{2}\right) + 2b \\ 39 &= \frac{36+12}{2} + 2b \\ b &= \frac{15}{2} \end{aligned} \tag{41}$$

The constant "*c*" can be found by using (35) for $n = 3$ ($m_1 = 3$)

$$\begin{aligned} p(3) = 43 &= \frac{3}{2}n^2 + \frac{15}{2}n + c = \frac{3}{2}3^2 + \frac{15}{2}3 + c \\ c &= 7 \end{aligned} \tag{42}$$

Finally, the closed-form expression for the dynamic range can be conjectured as

$$\hat{M}_{RSNS} = \frac{3}{2}m_1^2 + \frac{15}{2}m_1 + 7 \tag{43}$$

where $m_1 \geq 3$. This formula is checked against the computer results. For example, when $m_1 = 23$, $m_2 = 24$, and $m_3 = 25$, the dynamic range is 973 which fits (43).

## B. SPECIAL CASE: MODULI SPACING 2 APART

When the PRP channel numbers are 2 apart, the following results have been observed as shown in Table 4.2.

| $m_1$ | $m_2$ | $m_3$ | Dynamic Range | 1st Difference | 2nd Difference |
|---|---|---|---|---|---|
| 5 | 7 | 9 | 104 | | |
| 9 | 11 | 13 | 230 | 126 | |
| 13 | 15 | 17 | 404 | 174 | 48 |
| 17 | 19 | 21 | 626 | 222 | 48 |
| $\hat{M}_{RSNS} = \frac{3}{2}m_1^2 + \frac{21}{2}m_1 + 14$ | | | | | |

**Table 4.2: The Dynamic Range of** $m_1$, $m_2 = m_1 + 2$, **and** $m_3 = m_1 + 4$ **where** $m_1 \geq 5$

The first difference (e.g., 230 – 104 = 126) and second difference (e.g., 174 – 126 = 48) of the dynamic range of the 3-channel RSNS for $m_1$, $m_2 = m_1 + 2$, and $m_3 = m_1 + 4$ where $m_1 \geq 5$ are shown in Table 4.2. The relationship between the dynamic range differences can also be expressed as a 2nd degree polynomial

$$p(n) = an^2 + bn + c \tag{44}$$

where $n$ is the lower channel value and $a$, $b$, and $c$ are constant real numbers. Each set of the 3-moduli RSNS values shows a uniform increase of 4. Therefore, if the first set of channel values corresponds to (44), then the second set of channel values corresponds to

$$\begin{aligned} p(n+4) &= a(n+4)^2 + b(n+4) + c \qquad (45) \\ &= a(n^2 + 8n + 16) + bn + 4b + c \\ &= an^2 + 8an + 16a + bn + 4b + c \\ &= p(n) + 8an + 16a + 4b. \end{aligned}$$

The first difference of the sets ((45) – (44)) is

$$q(n) = p(n+2) - p(n) = 8an + 16a + 4b. \tag{46}$$

If the first difference of the first two sets is (46), the first set difference of the second two sets can be written as

$$\begin{aligned} q(n+4) &= 8a(n+4) + 16a + 4b \\ &= 8an + 32a + 16a + 4b \\ &= q(n) + 32a. \end{aligned} \tag{47}$$

The second difference of the sets are unique and

$$q(n+2) - q(n) = 32a. \tag{48}$$

The constant "*a*" can be found by using (48)

$$q(n+4) - q(n) = 32a = 48 \tag{49}$$

$$a = \frac{48}{32} = \frac{3}{2}$$

The constant "*b*" can be found by using (46) for $n = 5\ (m_1 = 5)$

$$\begin{aligned} p(n+4) - p(n) = 126 &= 8\left(\frac{3}{2}\right)5 + 16\left(\frac{3}{2}\right) + 4b \\ 126 &= \frac{120+48}{2} + 4b \\ b &= \frac{21}{2} \end{aligned} \tag{50}$$

The constant "*c*" can be found by using (44) for $n = 5\ (m_1 = 5)$

$$\begin{aligned} p(5) = 104 &= \frac{3}{2}n^2 + \frac{21}{2}n + c = \frac{3}{2}5^2 + \frac{21}{2}5 + c \\ c &= 14 \end{aligned} \tag{51}$$

Finally, the closed-form expression for the dynamic range can be conjectured as

$$\hat{M}_{RSNS} = \frac{3}{2}m_1^2 + \frac{21}{2}m_1 + 14 \tag{52}$$

where $m_1 \geq 3$. This formula is checked against the computer results. For example, when $m_1 = 25$, $m_2 = 27$, and $m_3 = 29$, the dynamic range is *1214* which fits (52).

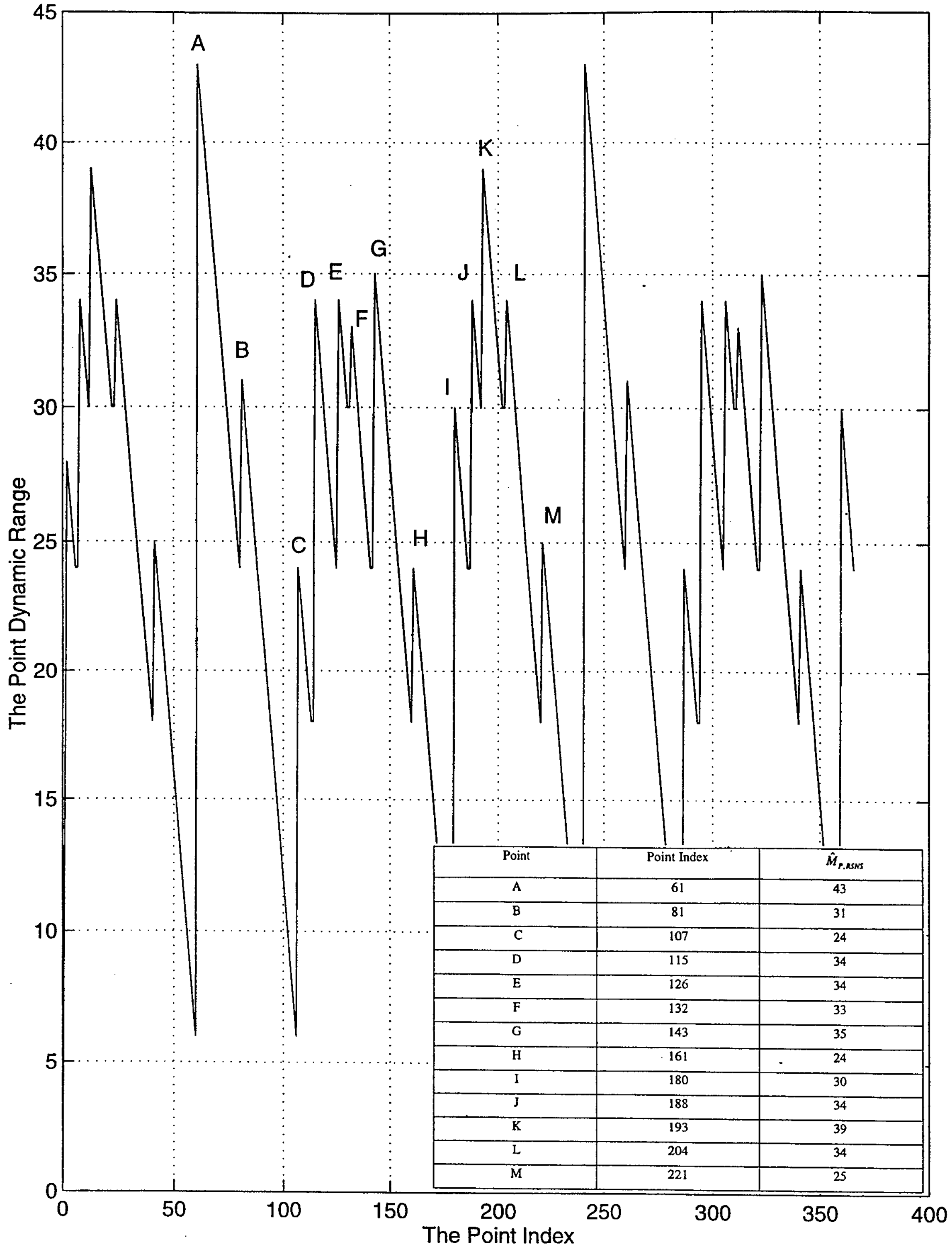
## C. GRAPHICAL ANALYSIS

The algorithm (**rsns.m**) also finds and displays the point dynamic range of the 3-channel RSNS as explained in Chapter II. The point dynamic range versus the point index for the channel values $m_1 = 3$, $m_2 = 4$, and $= 5$ are shown in Figures 4.1 through

4.6. The waveform with the points A, B, C, D, E, F, G, H, I, J, K, L, and M, that are shown in Figures 4.1 through 4.6, construct a special waveform that repeats itself every $\frac{P_{RSNS}}{2}$. This special waveform, the point dynamic range (PDR) waveform, gives the dynamic range starting at each point. The fundamental period of this particular 3-channel RSNS example is 360 by (16). So, this spike waveform repeats itself every 180 points.

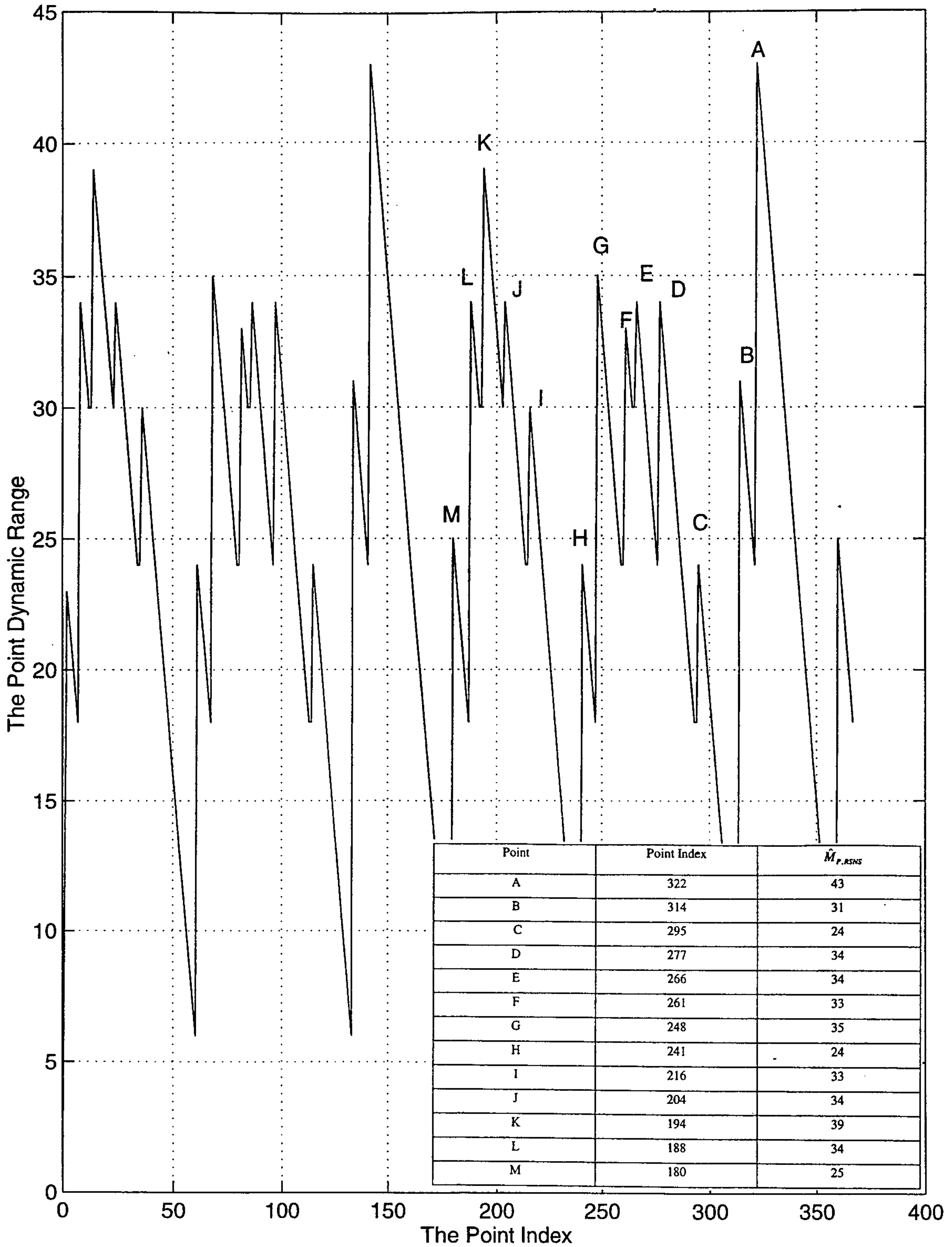
The maximum value of the point dynamic range is the dynamic range of the 3-channel RSNS. In this example, the dynamic range is 43. The first point index that yields the dynamic range is 61, 142, 8, 134, 180, and 81 for Figures 4.1 through 4.6, respectively. These values are the beginning point index of the dynamic range. The dynamic range point index values are also half of the fundamental period apart from each other.

The PDR waveform of the point dynamic range of the 3-channel RSNS will be the same for different shift values as shown in Figures 4.1 through 4.6. The beginning value of the dynamic range is different from each other. When one of the shift values is fixed, the other two shift value combinations of the PDR waveforms are opposite of each other. The PDR waveform of the Figures 4.1, 4.3, and 4.5 are the opposite of Figures 4.2, 4.4, and 4.6, respectively.



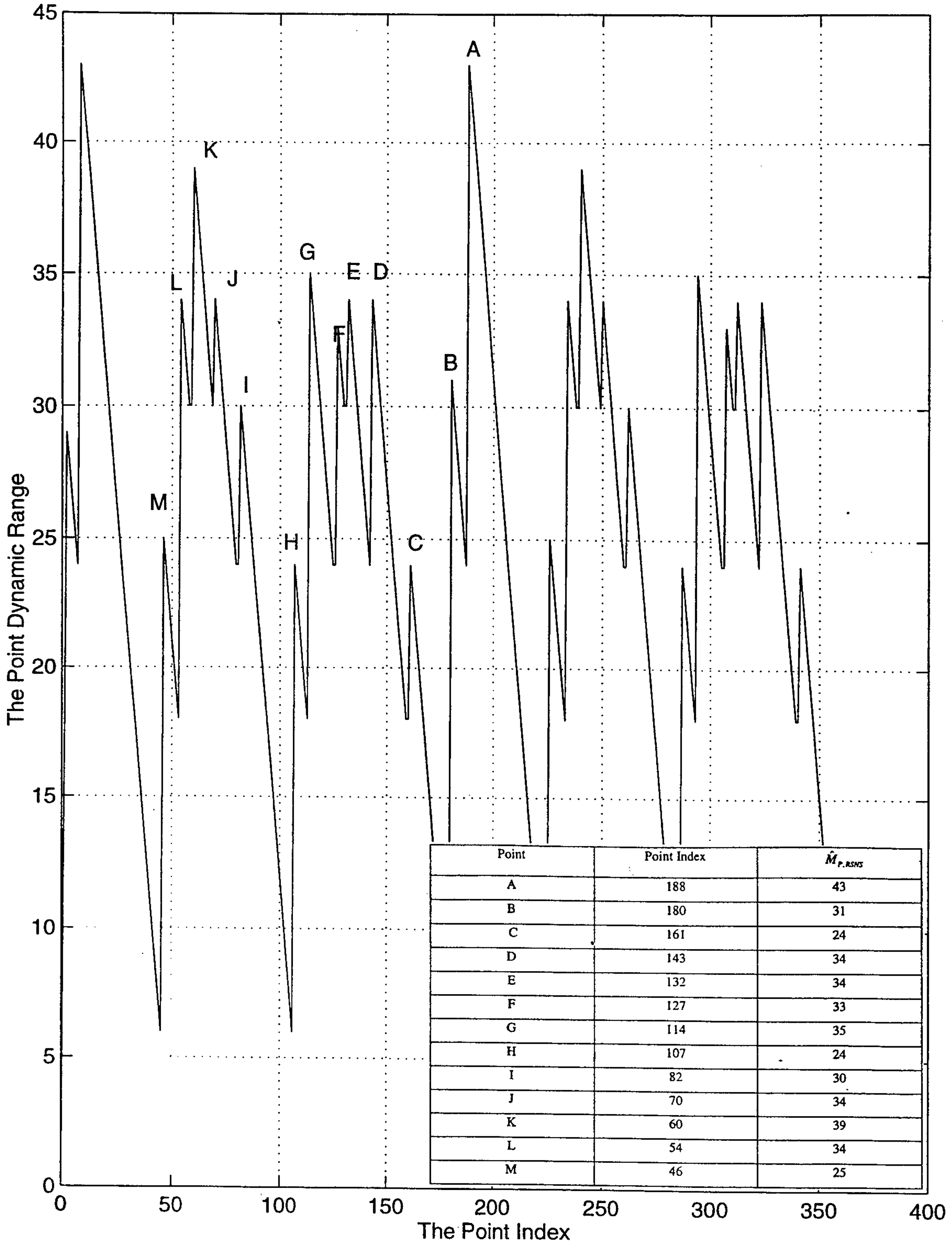
| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 61 | 43 |
| B | 81 | 31 |
| C | 107 | 24 |
| D | 115 | 34 |
| E | 126 | 34 |
| F | 132 | 33 |
| G | 143 | 35 |
| H | 161 | 24 |
| I | 180 | 30 |
| J | 188 | 34 |
| K | 193 | 39 |
| L | 204 | 34 |
| M | 221 | 25 |

**Figure 4.1: The Point Dynamic Range Waveform for** $m_1 = 3$, $s_1 = 0$, $m_2 = 4$, $s_2 = 1$, $m_3 = 5$, $s_3 = 2$.



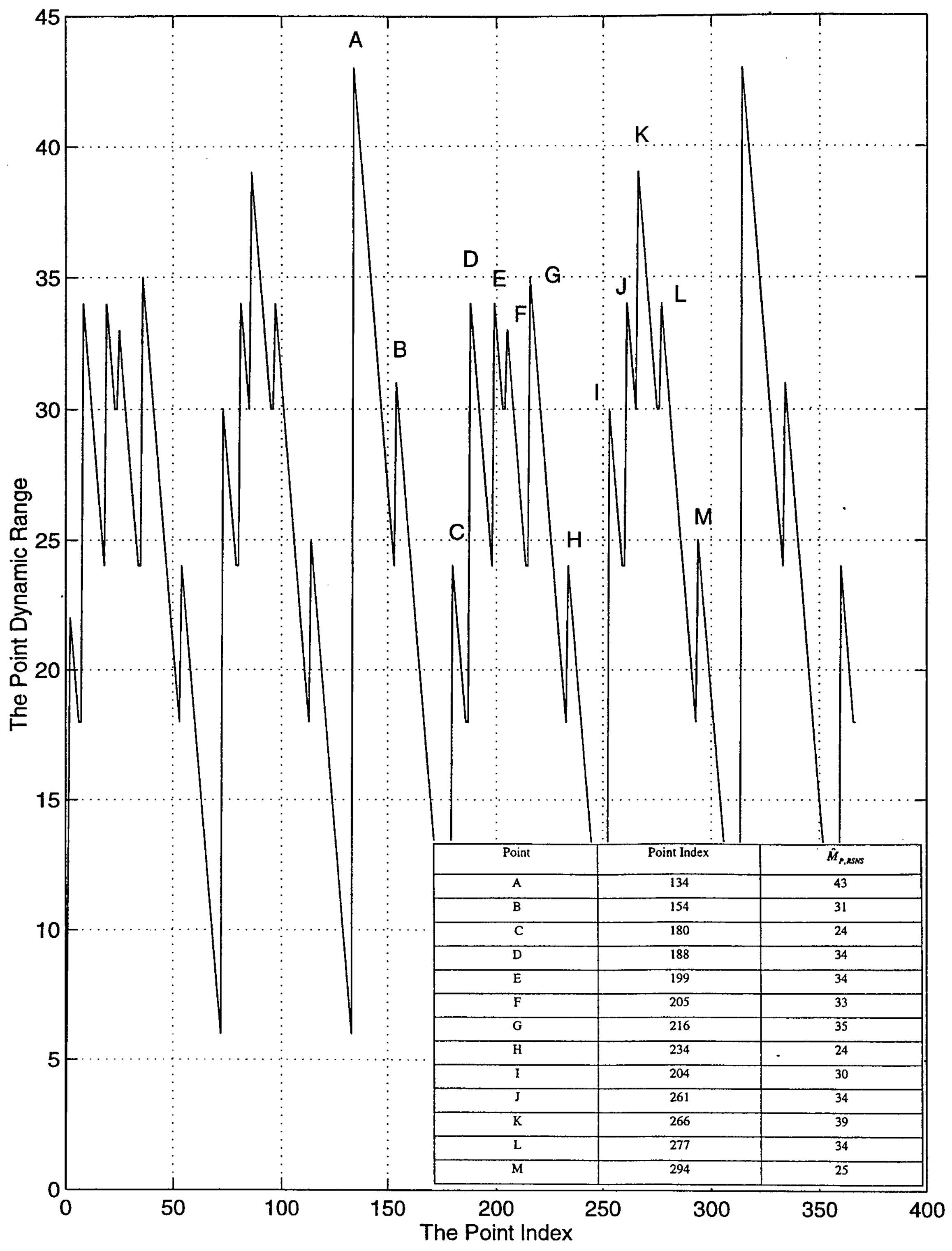
| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 322 | 43 |
| B | 314 | 31 |
| C | 295 | 24 |
| D | 277 | 34 |
| E | 266 | 34 |
| F | 261 | 33 |
| G | 248 | 35 |
| H | 241 | 24 |
| I | 216 | 33 |
| J | 204 | 34 |
| K | 194 | 39 |
| L | 188 | 34 |
| M | 180 | 25 |

**Figure 4.2: The Point Dynamic Range Waveform for** $m_1 = 3, s_1 = 0, m_2 = 4, s_2 = 2, m_3 = 5, s_3 = 1.$



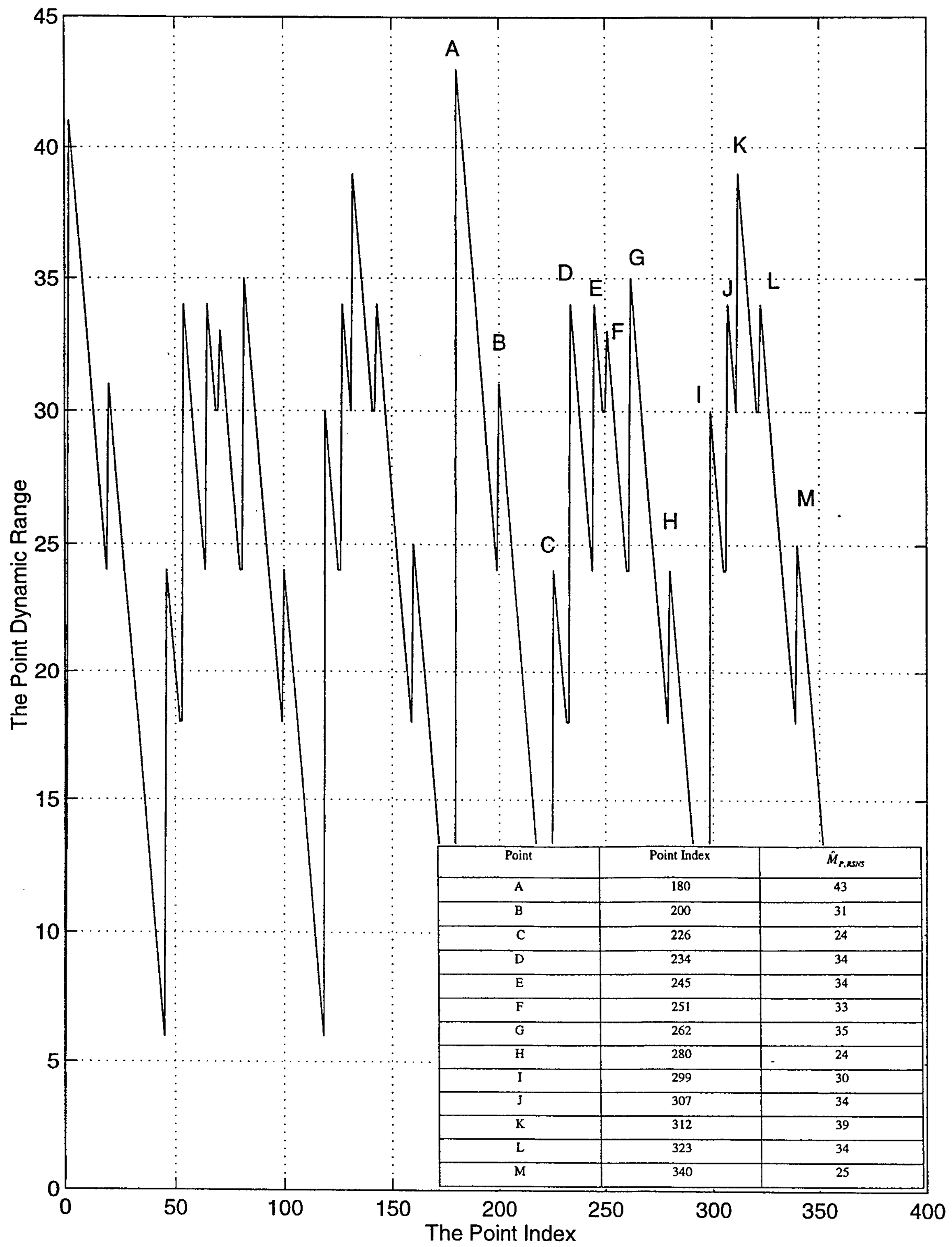
| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 188 | 43 |
| B | 180 | 31 |
| C | 161 | 24 |
| D | 143 | 34 |
| E | 132 | 34 |
| F | 127 | 33 |
| G | 114 | 35 |
| H | 107 | 24 |
| I | 82 | 30 |
| J | 70 | 34 |
| K | 60 | 39 |
| L | 54 | 34 |
| M | 46 | 25 |

**Figure 4.3: The Point Dynamic Range Waveform for** $m_1 = 3$, $s_1 = 1$, $m_2 = 4$, $s_2 = 0$, $m_3 = 5$, $s_3 = 2$.



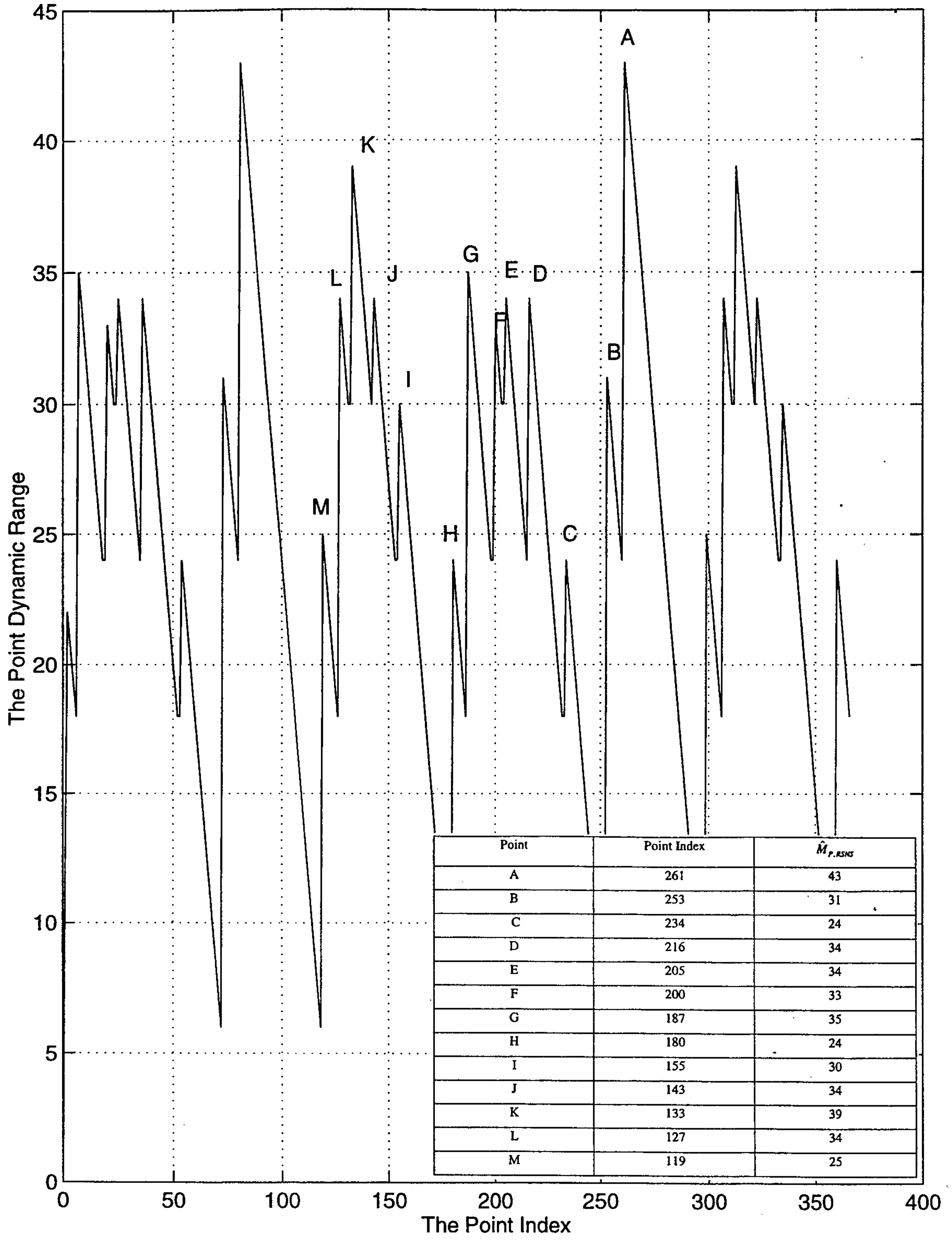
| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 134 | 43 |
| B | 154 | 31 |
| C | 180 | 24 |
| D | 188 | 34 |
| E | 199 | 34 |
| F | 205 | 33 |
| G | 216 | 35 |
| H | 234 | 24 |
| I | 204 | 30 |
| J | 261 | 34 |
| K | 266 | 39 |
| L | 277 | 34 |
| M | 294 | 25 |

**Figure 4.4: The Point Dynamic Range Waveform for** $m_1 = 3$, $s_1 = 1$, $m_2 = 4$, $s_2 = 2$, $m_3 = 5$, $s_3 = 0$.



| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 180 | 43 |
| B | 200 | 31 |
| C | 226 | 24 |
| D | 234 | 34 |
| E | 245 | 34 |
| F | 251 | 33 |
| G | 262 | 35 |
| H | 280 | 24 |
| I | 299 | 30 |
| J | 307 | 34 |
| K | 312 | 39 |
| L | 323 | 34 |
| M | 340 | 25 |

**Figure 4.5: The Point Dynamic Range Waveform for** $m_1 = 3$, $s_1 = 2$, $m_2 = 4$, $s_2 = 0$, $m_3 = 5$, $s_3 = 1$.



| Point | Point Index | $\hat{M}_{P,RSNS}$ |
|---|---|---|
| A | 261 | 43 |
| B | 253 | 31 |
| C | 234 | 24 |
| D | 216 | 34 |
| E | 205 | 34 |
| F | 200 | 33 |
| G | 187 | 35 |
| H | 180 | 24 |
| I | 155 | 30 |
| J | 143 | 34 |
| K | 133 | 39 |
| L | 127 | 34 |
| M | 119 | 25 |

**Figure 4.6: The Point Dynamic Range Waveform for** $m_1 = 3, s_1 = 2, m_2 = 4, s_2 = 1, m_3 = 5, s_3 = 0$.

# V. APPLICATION OF THE RSNS TO FOLDING ANALOG-TO-DIGITAL CONVERTERS

## A. BACKGROUND FOR THE FOLDING ADC

Analog to digital converters (ADCs) are critical building blocks in a wide range of hardware, from radar and electronic warfare systems to multimedia based personal computer and work stations. The best-known architecture for high-speed ADCs is the flash converter structure [15]. In this structure an array of comparators amplitude analyze the input voltage with a set of increasing reference voltages. The comparator outputs represent the input signal in a thermometer code, which can be easily converted into a gray or binary code. This architecture requires $(2^N - 1)$ comparators to achieve $N$-bit resolution, which makes it difficult to achieve a high resolution while maintaining, at the same time, a large analog bandwidth, a low power dissipation, and a small die area.

The need constantly exists for converters with higher resolution, faster conversion speeds, and lower power dissipation. To reduce the number of power consuming components, high performance ADCs employ a parallel configuration of analog folding circuits to symmetrically fold the input signal prior to quantization by high speed comparators (analog preprocessing) [6-9]. Recently a SNS preprocessing technique has been described that can easily be incorporated into the established techniques to provide an enhanced resolution capability with a fewer comparators loaded in parallel. The approach is based on preprocessing the analog signal with symmetrical number systems (SNS and OSNS). The preprocessing is used to decompose the analog amplitude analyzer operation into $N$ sub-operations (channels) which are of smaller computational complexity. Each sub-operation symmetrically folds the analog signal with folding period equal to the channels or twice the channels. Thus, each sub-operation only requires a precision in accordance with that modulus. A much higher resolution is achieved after the $N$ different channels are used and the results of these low precision sub-operations are combined. Increasing the resolution of these architectures can be accomplished by incorporating more efficient encoding. One of the important factors which helps to measure the efficiency of the SNS encoding is the dynamic range. The maximum number of consecutive vectors without a redundancy. The other factor is the total number of comparators, which are required for the each encoding. The 7-bit OSNS ADC block diagram is shown in Figure 5.1. The problem with the SNS and OSNS is the encoding errors. The transfer function of 7-bit OSNS and the encoding errors for 0% LSB decimation width is shown in Figure 5.2.

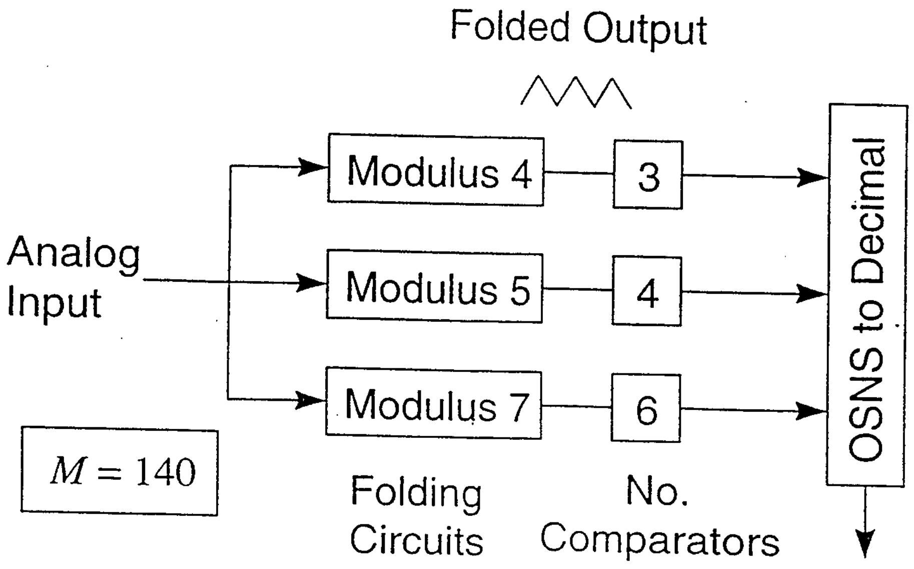


**Figure 5.1: The 7-bit OSNS Block Diagram**

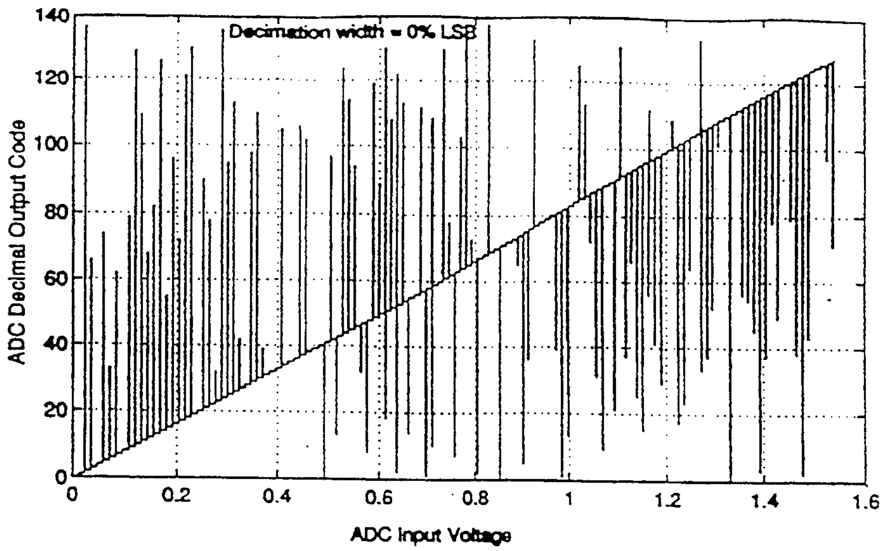


**Figure 5.2: The Transfer Function with Isolation Width = 0% LSB for 7-bit OSNS**

## B. IMPLEMENTATION ISSUES

To demonstrate the efficiency of the RSNS preprocessing a unipolar 5-bit ADC with $m_1 = 3$, $m_2 = 4$, and $m_3 = 5$ is considered. The dynamic range of this 3-channel RSNS is 43. A schematic diagram of this folding ADC is shown in Figure 5.3.

The input signal is folded in parallel with each folding period, $P_{RSNS}$ equal to $2Nm$, where $N$ is the number of channels and $m$ is the modulus (channel value). The output from each folding circuit is quantitized with $m$ comparators. For this simulation, there will be 3 + 4 + 5 = 12 comparators total and a maximum 5 of them will load in parallel. The comparator threshold levels are adjusted to midlevel quantize the folded output waveform. One interconnecting single folding and comparator stage is shown in Figure 5.4 and the corresponding values of the resistors, current and voltage sources are shown in Table 5.1. The encoder to a binary representation for a high-resolution digital output converts the comparator outputs (the RSNS representation of the input signal).

Each RSNS folding circuit must symmetrically fold the input signal at least $\frac{\hat{M}_{RSNS}}{2Nm}$ number of times. High performance folding circuits typically consists of several identical but independent stages inter-connected in parallel with one stage for each required fold. An additional stage is sometimes used to register the input voltages in DC.

$V_{ref}$ is a supplied reference voltage to each stage equal to the input voltage point at which a fold (peak) is desired. Also shown are the analog input node and the folded output node. For this 5-bit example, a LSB code width of *12mv* with a full-scale voltage $(2^6\text{-}1)$ x $12mv = 0.756$ v is chosen. The folding period for the modulus $m_i$ channel is

$$T_{m_i} = 2Nm_i \times (LSB\ Code\ Width)\ v \tag{53}$$

where $N$ is the number of modulus and $m_i$ is the value of the modulus and it goes from 1 to $N$. The first required reference voltages ($V_{ref(1)}$- above zero and $V_{ref(0)}$- below zero) are

$$V_{ref(1)} = \frac{T_{m_i}}{2} + (s_i \times (LSB\ Code\ Width))\ v \tag{54}$$

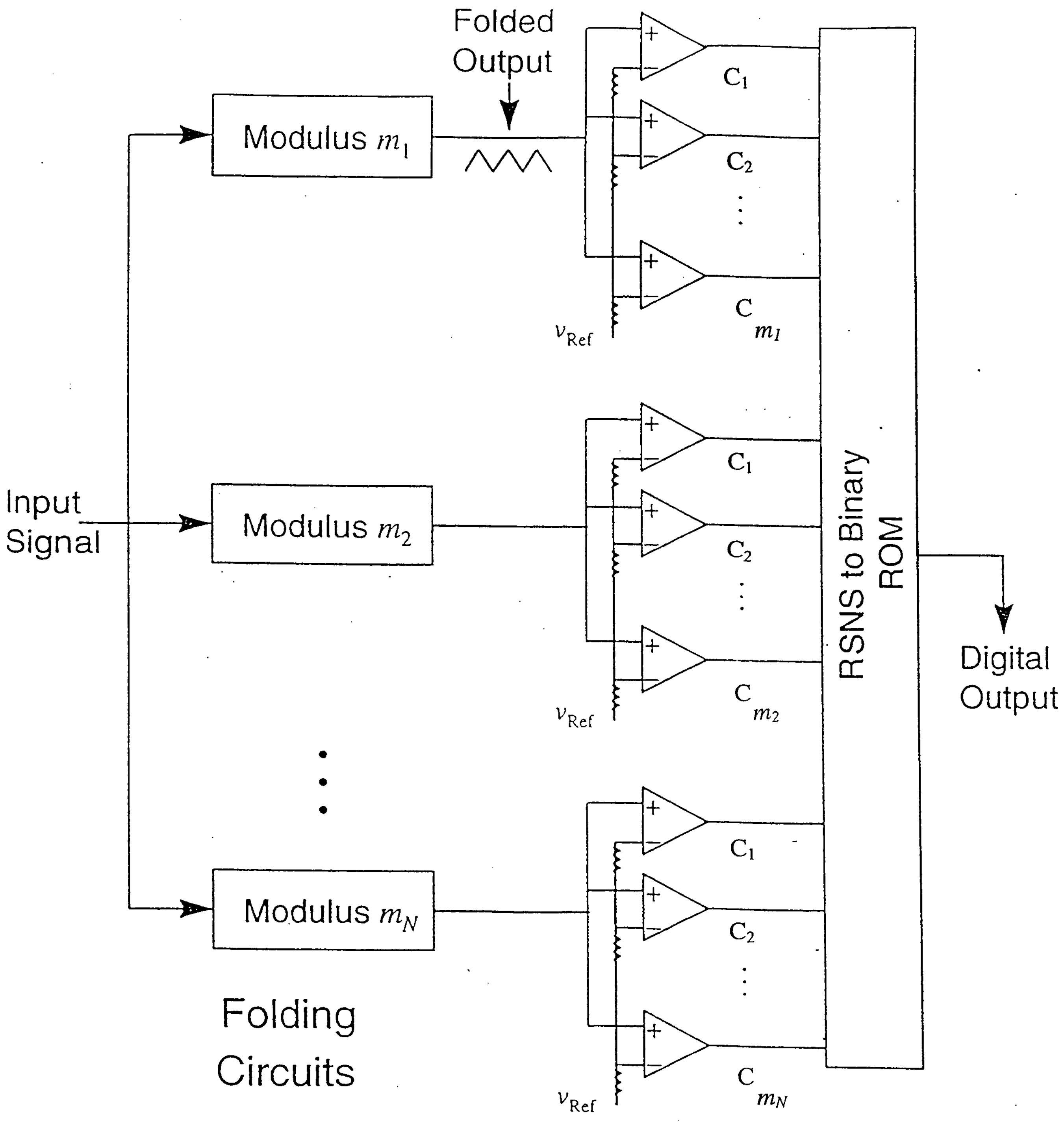


**Figure 5.3: The 5-bit RSNS ADC Block Diagram**

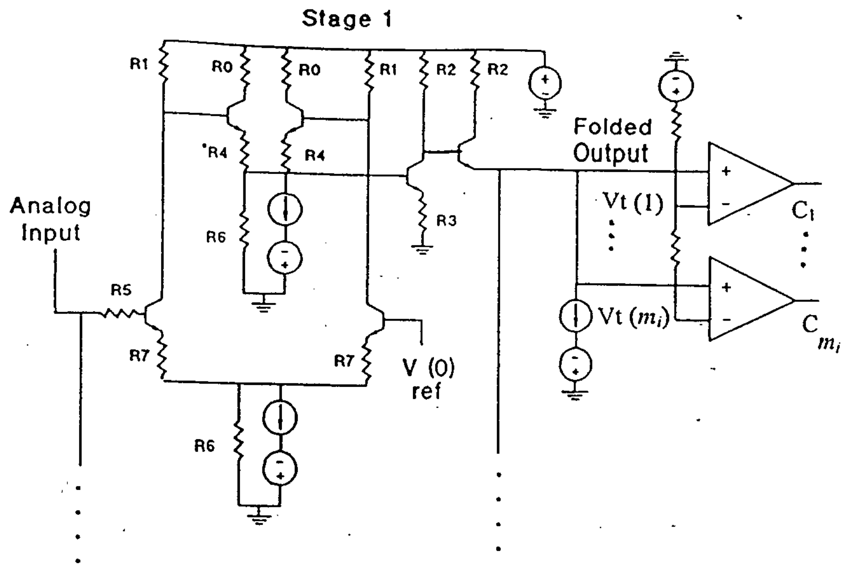


**Figure 5.4: One Interconnecting Single Folding and Comparator Stages of the 5-bit RSNS**

| | | |
|---|---|---|
| Resistor Design Values | R0 | 1300 Ω |
| | R1 | 10,000 Ω |
| | R2 | 900 Ω |
| | R3 | 400 Ω |
| | R4 | 300 Ω |
| | R5 | 300 Ω |
| | R6 | 250 Ω |
| | R7 | 250 Ω ( mod-3) - 300 Ω (mod-4) – 350 Ω (mod-5) |
| Voltage Sources Design Values | V1 | 15 v |
| | V2 | 15 v |
| | V3 | 15 v |
| Current Sources Design Values | I1 | 0.001 A |
| | I2 | 0.001 A |

**Table 5.1: The Design Values of the 5-bit RSNS** ($m_1$ = 3, $m_2$ = 4, $m_3$ =5)

$$V_{ref(0)} = -\frac{T_{m_i}}{2} + (s_i \times (LSB\ Code\ Width))\ v \qquad (55)$$

where $T_{m_i}$ is the folding period and $s_i$ is the shift values of the modulus $m_i$. The other required reference voltages are

$$V_{ref(N)} = \frac{T_{m_i}}{2} + (s_i \times (LSB\ Code\ Width)) + NT_{m_i} \quad v \tag{56}$$

where $N$ is the $N^{th}$ reference voltages and lies between {0, 1, 2, …} for the modulus $m_i$.

The folding period by (53) for the $m_1 = 3$ is $T_3 = 2$ x $3$ x $3$ x $12mv = 216\ mv$. That is, the folding period is scaled to the modulus. For $m_1 = 3$, the shift value is taken $s_1 = 0$. The first required reference voltage, above zero is then $V_{ref(1)} = \frac{T_3}{2} = 108\,mv$ by (54). The reference voltage, below zero is $V_{ref(0)} = -\frac{T_3}{2} = -108\,mv$ by (55). The other required reference voltages for $m_1 = 3$ are calculated by (56) and shown in Table 5.2.

The folding period by (53) for the $m_2 = 4$ is $T_4 = 2$ x $3$ x $4$ x $12mv = 288\ mv$. That is, the folding period is scaled to the modulus. For $m_2 = 4$, the shift value is taken $s_2 = 1$. The first required reference voltage, above zero is then $V_{ref(1)} = \frac{T_4}{2} + (s_1 \times (LSB\ Code\ Width)) = 156mv$ by (54). The reference voltage, below zero is $V_{ref(0)} = -\frac{T_4}{2} + (s_2 \times (LSB\ Code\ Width)) = -132mv$ by (55). The other required reference voltages for $m_2 = 4$ are calculated by (56) and shown in Table 5.2.

The folding period for the $m_3 = 5$ is $T_5 = 2$ x $3$ x $5$ x $12mv = 360\ mv$ by (53). That is, the folding period is scaled to the modulus. For $m_3 = 5$, the shift value is taken $s_3 = 2$. The first required reference voltage, above zero is then $V_{ref(1)} = \frac{T_5}{2} + (s_3 \times (LSB\ Code\ Width)) = 204mv$ by (54). The reference voltage, below zero is $V_{ref(0)} = -\frac{T_5}{2} + (s_3 \times (LSB\ Code\ Width)) = -156mv$. The other required reference voltages for $m_3 = 5$ are calculated by (56) and shown in Table 5.2.

| Fold Number | Modulus 3 ($T_3$=*0.216*) | Modulus 4 ($T_4$=*0.288*) | Modulus 5 ($T_5$=*0.360*) |
|---|---|---|---|
| 0 | -0.108 | -0.132 | -.0156 |
| 1 | 0.108 | 0.156 | 0.204 |
| 2 | 0.324 | 0.444 | 0.564 |
| 3 | 0.540 | 0.732 | 0.924 |
| 4 | 0.756 | 1.020 | 1.284 |
| 5 | 0.972 | 1.308 | 1.644 |
| 6 | 1.188 | 1.596 | - |
| 7 | 1.404 | - | - |
| 8 | 1.620 | | |

**Table 5.2: The Reference Voltages of the 5-bit RSNS Design**

A SPICE simulation is used to generate the folding waveforms and comparator outputs states. A logic map (or an algorithm (**spice.m**) to give a decimal output as shown in the Appendix processes the sampled folding waveforms and comparator states. The steady-state folding waveforms for each channel are shown in Figures 5.5, 5.6, and 5.7. These folding waveforms are generated using a step size of $\Delta v = 0.001\ v$ in the SPICE simulation.

The comparator threshold voltages $V_t$ for each modulus derived from the folding waveforms to mid-level quantize the input signal into RSNS format. The threshold levels occur at the code transition points and are tailored to the particular folding waveform being instrumented. The comparator threshold voltages for the 5-bit RSNS are shown in Table 5.3. Note the threshold values are not uniformly spaced. The comparator outputs are shown in Figures 5.5, 5.6 and 5.7 with their corresponding folded waveform for 5-bit RSNS.

| Comparator Number | Modulus-3 | Modulus-4 | Modulus-5 |
|---|---|---|---|
| 1 | 5.4097 | 5.2994 | 5.2234 |
| 2 | 5.6748 | 5.5169 | 5.4045 |
| 3 | 5.9452 | 5.7408 | 5.5946 |
| 4 | - | 5.9668 | 5.7878 |
| 5 | - | - | 5.9818 |

**Table 5.3: The Comparator Threshold Voltages for the 5-bit RSNS Design**

## C. TRANSFER FUNCTION

To obtain a more convenient digital output, the RSNS encoded signals from each channel are recombined. The comparator outputs from each channel are decoded using a logic block or programmable logic array (PLA). The RSNS-to-decimal mapping function

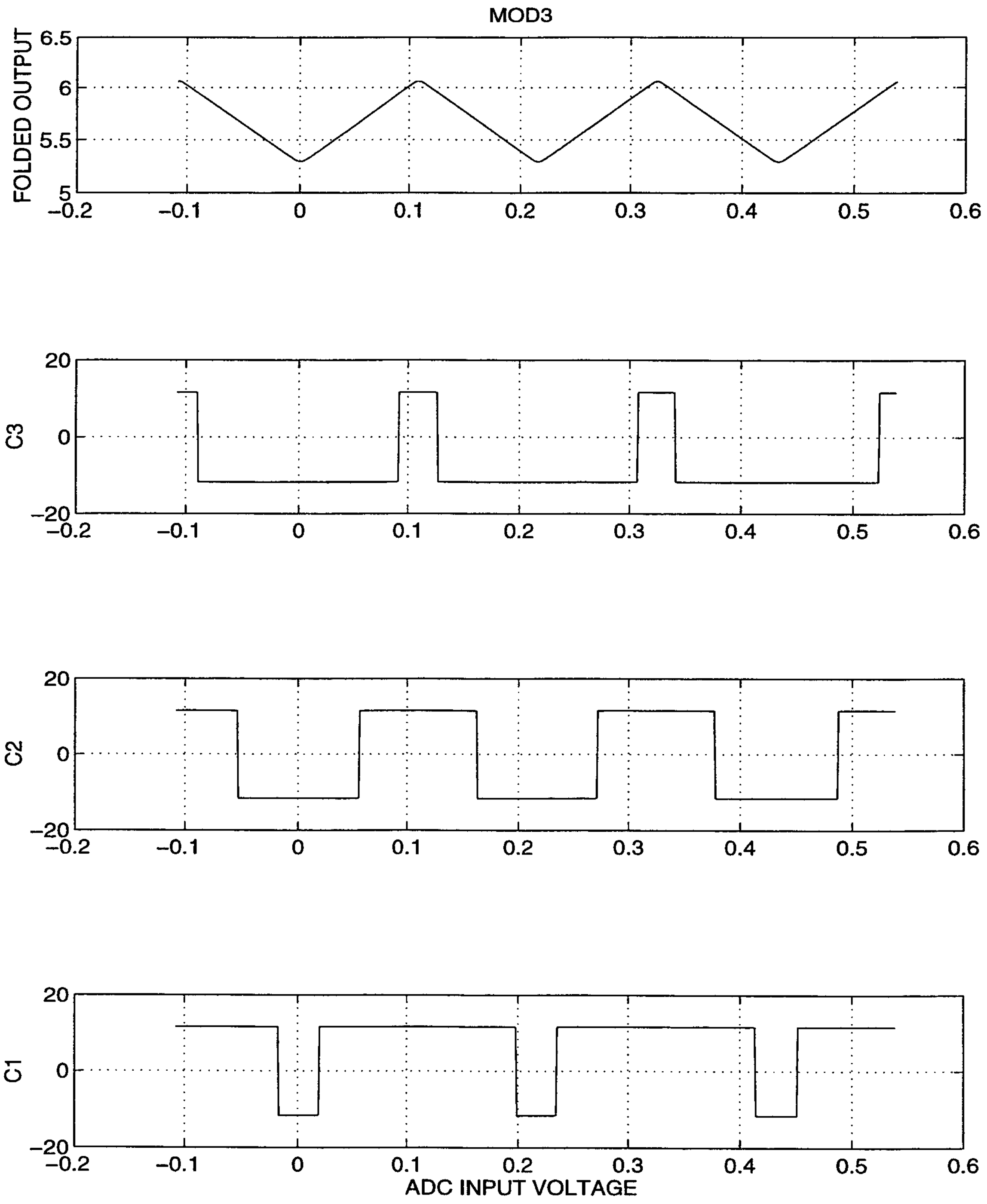


**Figure 5.5: Folding Output Waveform/Comparator Outputs for the RSNS $m_1 = 3$**

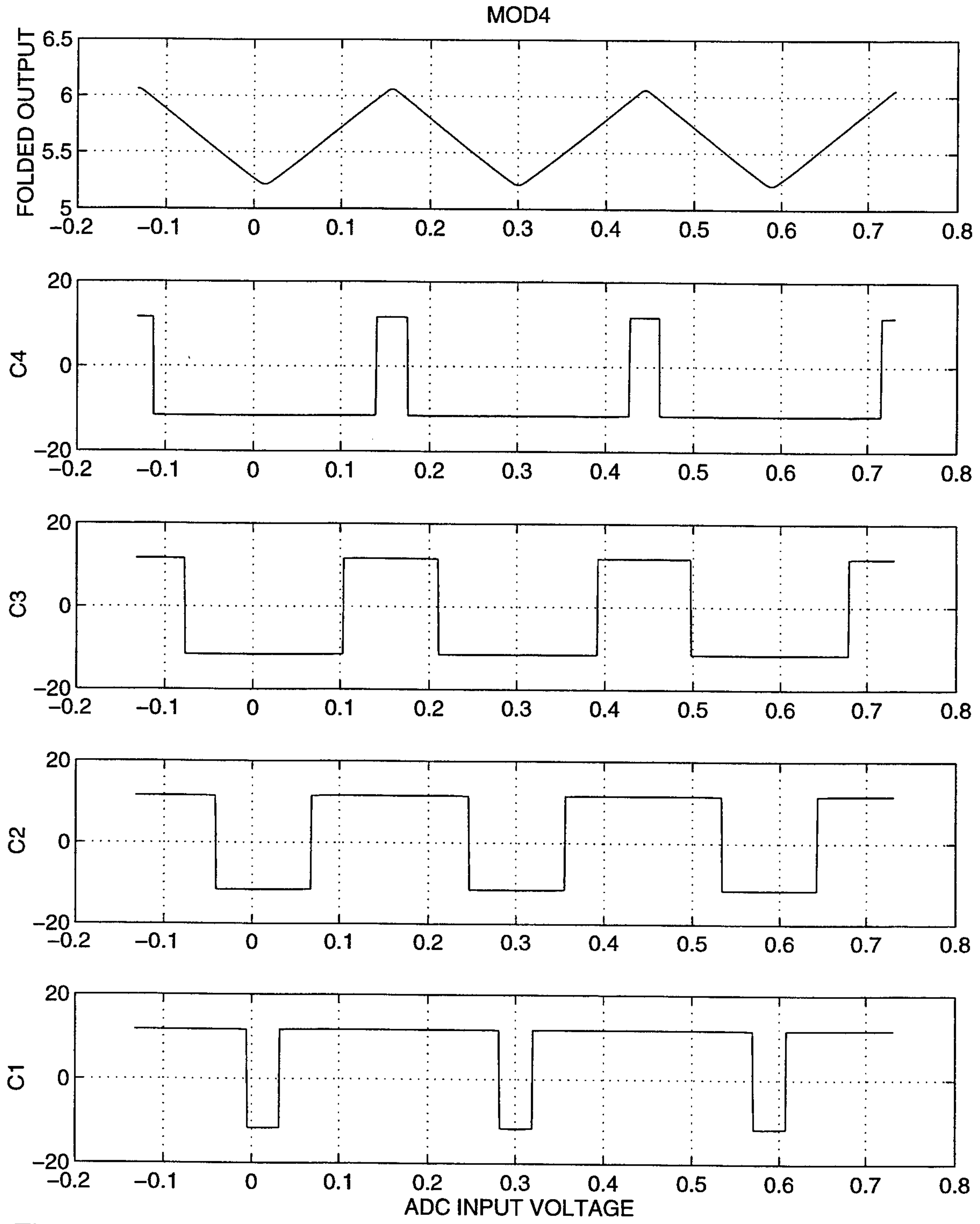


**Figure 5.6: Folding Output Waveform/Comparator Outputs for the RSNS $m_2 = 4$.**

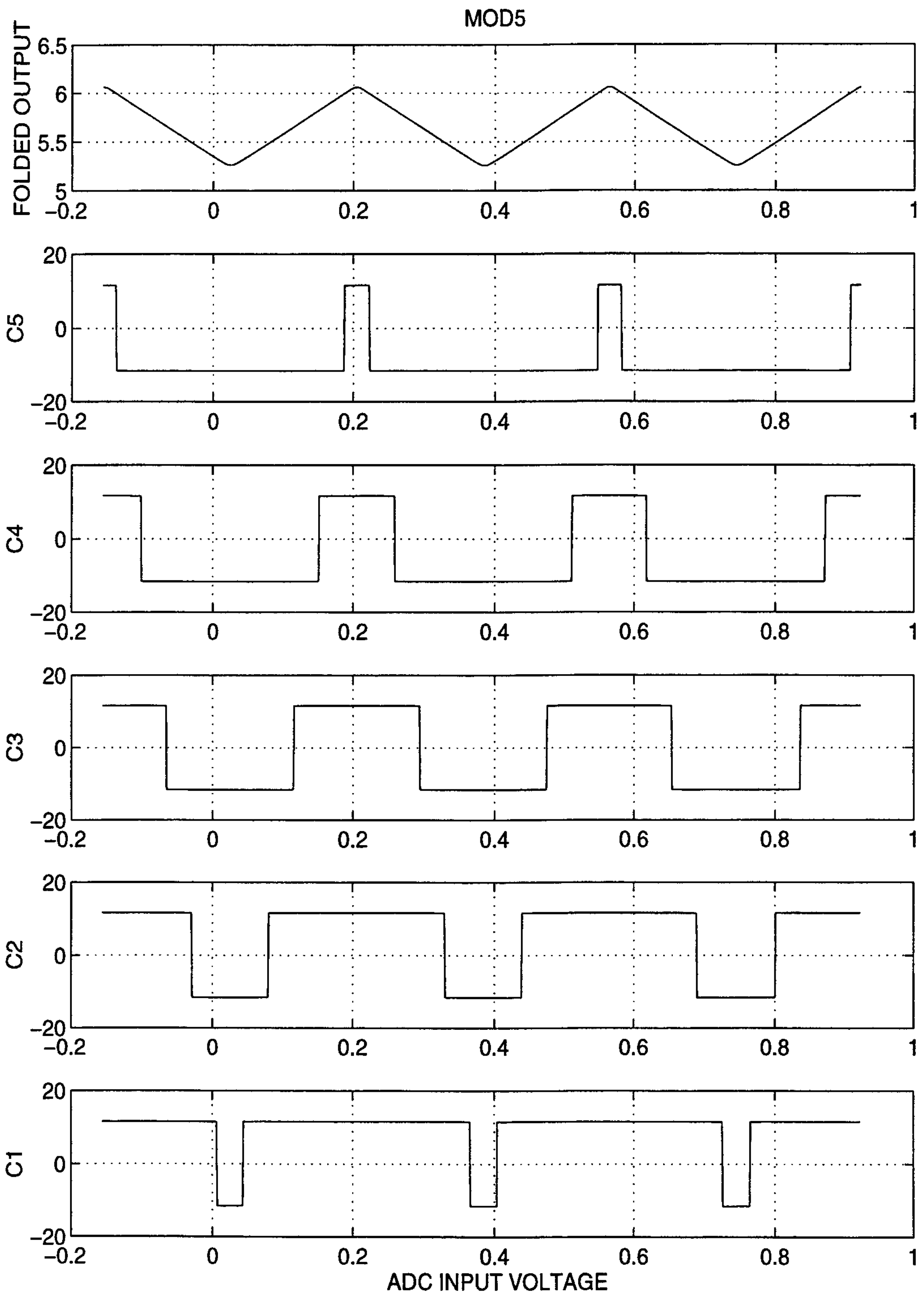


**Figure 5.7: Folding Output Waveform/Comparator Outputs for the RSNS** $m_3 = 5$.

for the 5-bit RSNS design is shown in Table 5.4 where each entry indicates the number of comparators in the ON state.

| The Point Index | RSNS Moduli | | |
|---|---|---|---|
| | 3 | 4 | 5 |
| 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 2 | 1 | 1 | 0 |
| 3 | 1 | 1 | 1 |
| 4 | 2 | 1 | 1 |
| 5 | 2 | 2 | 1 |
| 6 | 2 | 2 | 2 |
| 7 | 3 | 2 | 2 |
| 8 | 3 | 3 | 2 |
| 9 | 3 | 3 | 3 |
| 10 | 2 | 3 | 3 |
| 11 | 2 | 4 | 3 |
| 12 | 2 | 4 | 4 |
| 13 | 1 | 4 | 4 |
| 14 | 1 | 3 | 4 |
| 15 | 1 | 3 | 5 |
| 16 | 0 | 3 | 5 |
| 17 | 0 | 2 | 5 |
| 18 | 0 | 2 | 4 |
| 19 | 1 | 2 | 4 |
| 20 | 1 | 1 | 4 |

**Table 5.4: The 5-bit RSNS-to-Decimal Mapping**

The RSNS ADC architecture is simulated using a sampling period of $\Delta v = 0.001\ v$ with the corresponding transfer function shown in Figure 5.8. The LSB code width is assumed *12 mv* for this simulation. Therefore, the dynamic range lies between the input voltages of $0.012 \times 61 = 0.732\ v$ and $0.012 \times 103 = 1.236\ v$. The entire graph. Figure 5.2 lies in the dynamic range. This corresponds to the portion of Figure 5.8 where the dynamic range lies from *0.732 v* to *1.236 v*. When the sampling period is decreased to $\Delta v = 0.0004\ v$, the transfer function does not change. Note that the encoding errors typically present in the SNS and OSNS ADCs do not exist. This clearly demonstrates the robustness of the RSNS encoding (Gray code properties). For this system, channel and shift values, the dynamic range lies between the point indexes 61 and 103 as shown in Table 2.4.

The comparator states of this architecture are simulated to the output decimal value by using the algorithm, **trans.m**. The beginning point index (61) of the dynamic range is assigned the decimal value 0. The dynamic range point indexes are assigned the decimal values from 0 to 42. The point indexes within the dynamic range are assigned the decimal values starting from 0 to 42. Then, starting from the point index 60 to 0, the decimal values are assigned backwards. If the point index of the discrete states is a redundancy within the dynamic range, it is assigned the corresponding decimal value. If not, it receives a new decimal value starting from -1. Finally, the same procedure is applied to the point index starting from 104, but the values of the decimal codes start from 43. The dynamic range (useful code combinations) and the redundancies (ambiguities) can be seen from Figure 5.8.

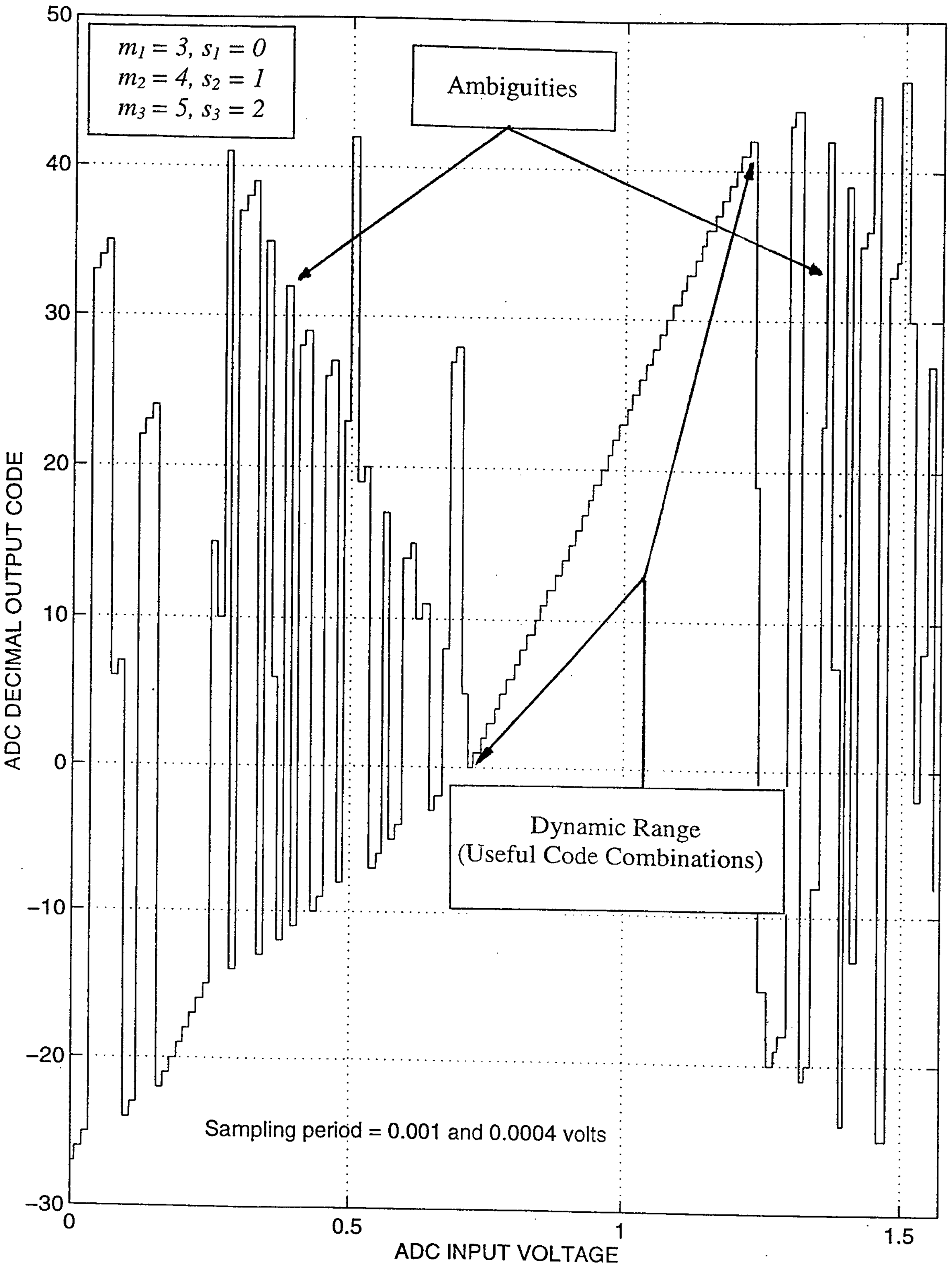


**Figure 5.8: The 5-bit RSNS ADC Transfer Function** ($\Delta v$ = *0.001* **and** $\Delta v$ = *0.0004 v*)

# VI. CONCLUDING REMARKS

The comparison of the RSNS with the other SNS formulations is shown in Table 6.1. The folding period for the RSNS is a factor of *2N* greater then the SNS and *N* greater

| Scheme | Folding Periods | Number of Comparators Per Channel | Dynamic Range |
|---|---|---|---|
| SNS | $m_i$ | $\left\lfloor \frac{m_i}{2} \right\rfloor$ | $\prod_{i=1}^{N} m_i$ |
| SNS (all moduli odd) | $m_i$ | $\left\lfloor \frac{m_i}{2} \right\rfloor$ | $\min\left\{\frac{1}{2}\prod_{l=1}^{j} m_{i_l} + \frac{1}{2}\prod_{l=j+1}^{N} m_{i_l}\right\}$ |
| SNS (one moduli even) | $m_i$ | $\left\lfloor \frac{m_i}{2} \right\rfloor$ | $\min\left\{\frac{m_1}{2}\prod_{l=2}^{j} m_{i_l} + \prod_{l=j+1}^{N} m_{i_l}\right\}$ |
| OSNS | $2m_i$ | $m_i - 1$ | $\prod_{i=1}^{N} m_i$ |
| RSNS (2 moduli) | $2Nm_i$ | $m_i$ | $4m_1 + 5m_2 - 5$ (moduli 1 or 2 apart) |
| RSNS (2 moduli) | $2Nm_i$ | $m_i$ | $4m_1 + 5m_2 - 2$ (moduli 3 or more apart) |
| RSNS (3 moduli) | $2Nm_i$ | $m_i$ | $\frac{3}{2}m_1^2 + \frac{15}{2}m_1 + 7$ (moduli 1 apart) |
| RSNS (3 moduli) | $2Nm_i$ | $m_i$ | $\frac{3}{2}m_1^2 + \frac{21}{2}m_1 + 14$ (moduli 2 apart) |

**Table 6.1: The Summary of the SNS Systems**

then the OSNS. The number of comparators per channel for the RSNS is the maximum of all the SNS systems. The dynamic range for the RSNS is less than the OSNS, but greater than the SNS.

The essential contribution of this thesis is defining a new symmetrical number system, robust symmetrical number system (RSNS) and investigating its properties. High Speed Analog-to-Digital Converters (ADCs) for 2-channel and 3-channel are summarized in Tables 6.2 and 6.3, respectively.

| Number System | Channel (Modulus) | Dynamic Range | Folding Period | Number of Comparators |
|---|---|---|---|---|
| **SNS** | 4 | 7 | 4 | 2 |
|  | 5 |  | 5 | 2 |
| **OSNS** | 4 | 20 | 8 | 3 |
|  | 5 |  | 10 | 4 |
| **RSNS** | 4 | 21 | 16 | 4 |
|  | 5 |  | 20 | 5 |

**Table 6.2: 2-Channel (modulus) Summary of the Number Systems for $m_1 = 4$ and $m_2 = 5$**

The dynamic range of the 2-channel RSNS is greater than the SNS and OSNS, but it has an increased number of comparators. For the 3-channel RSNS, although the dynamic range is greater than the SNS, it is less than the OSNS. The RSNS has Gray code properties in its behavior. Gray code properties mean only one comparator threshold is crossed at a time across parallel channels (moduli). This property of the RSNS eliminates the possibility of large errors at the code transition points and the requirements for interpolation signal processing.

Using SPICE, explained in Chapter V, simulates the 3-channel RSNS for specific channel and shift values. From the simulation results, it can be seen that the RSNS does not have any encoding errors. This property of the RSNS makes itself an attractive and useful number system in High Speed ADCs.

| Number System | Channel (Modulus) | Dynamic Range | Folding Period | Number of Comparators |
|---|---|---|---|---|
| **SNS** | 3 | 11 | 3 | 1 |
| | 4 | | 4 | 2 |
| | 5 | | 5 | 2 |
| **OSNS** | 3 | 60 | 6 | 2 |
| | 4 | | 8 | 3 |
| | 5 | | 10 | 4 |
| **RSNS** | 3 | 43 | 18 | 3 |
| | 4 | | 24 | 4 |
| | 5 | | 30 | 5 |

**Table 6.3: 3-Channel (modulus) Summary of the Number Systems for** $m_1 = 3, m_2 = 4$ **and** $m_3 = 5$

Future efforts will attempt to resolve the following important questions.

- How does the shift effect the beginning point index of the dynamic range for any RSNS configuration?
- Can a general closed-form expression be developed for the 3-channel and *N*-channel RSNS?

# APPENDIX

## A . DESCRIPTION OF THE COMPUTER ALGORITHM DEVELOPED

The programming in this study is accomplished using the MATLAB software package. The main program, which is used to find the properties of the RSNS, is the **rsns.m**. The program **spice.m** and **trans.m** are used to plot the SPICE simulation.

The program is **rsns.m** finds and displays the dynamic range, fundamental period and beginning-ending point index of the discrete states without redundancy of any number of RSNS channels. The channel and shift values must be entered to run the program. This program also finds and displays the point dynamic range of the RSNS.

The program **spice.m** reads the folding waveform and comparator outputs data and displays the folding waveform and comparator outputs for each channel (modulus) of the RSNS by using SPICE simulation.

The program **trans.m** also reads the folding waveform and comparator output data which are calculated by using SPICE simulation. The algorithm, **trans.m** finds the decimal output codes for each comparator outputs of the RSNS and displays the transfer function of it. The dynamic range and its location can be seen in this algorithm display (figure).

## B. PROGRAM LISTING

### 1. RSNS Properties (rsns.m)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% rsns.m                                                                     %
% Modified July 05th 1997                                                    %
% This program finds the dynamic range, point dynamic range, fundamental period %
% and beginning-ending values of ambiguity vectors for N-channel RSNS          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
% Enter the channel number and the values of them
disp('This program finds the maximal strings of non-redundant vectors');
disp('for the N channel ROBUST SNS');
% Define a variable for the channel number
chanum=input('Enter the Number of Channels for ROBUST SNS >> ');

% Find the fundamental period with respect to the entered channel and shift values
period=1;
for i=1:chanum
   m(i)=input(['Enter ' int2str(i) '.Channel Value >> ']);
   period=LCM(period,(2*chanum*m(i)));
end
nsearch=period+30;
prompt='y';

% Enter the corresponding shift values
while (prompt=='y')|(prompt=='Y')
for i=1:chanum
   s(i)=input(['Enter ' int2str(i) '. Channel Shift Value >> ']);
end
```

```
% Initiate the variables to zero
   i=0;
   ii=0;
   j=0;
   jj=0;
   k=0;
   % Define a variable for the fundamental period
   funper=0;
   % Define a variable for the dynamic range
   dynrange=0;

   % Define the waveform for the entered channel and shift values and put the data in
   % matrix "g"
   for r=1:chanum
         % Define the matrix "mm"  to put the entered channel and shift values
         mm(r,[1 2])=[m(r) s(r)];
         %  Use the definition of RSNS
         for i=1+s(r):chanum*m(r)+s(r)
            g(r,i)=floor((i-s(r))/chanum);
         end
         for i=chanum*m(r)+1+s(r): 2*chanum*m(r)+s(r)
            g(r,i)=floor((2*chanum*m(r)+chanum-i+s(r)-1)/chanum);
         end
% Extent enough the data of the folding waveforms of the RSNS to find the dynamic
% range and fundamental period
g(r,2*chanum*m(r)+s(r)+1:4*chanum*m(r)+s(r))=g(r,1+s(r):2*chanum*m(r)+s(r));
g(r,4*chanum*m(r)+s(r)+1:8*chanum*m(r)+s(r))=g(r,1+s(r):4*chanum*m(r)+s(r));
g(r,8*chanum*m(r)+s(r)+1:16*chanum*m(r)+s(r))=g(r,1+s(r):8*chanum*m(r)+s(r));
```

```
g(r,16*chanum*m(r)+s(r)+1:32*chanum*m(r)+s(r))=g(r,1+s(r):16*chanum*m(r)+s(r));
g(r,32*chanum*m(r)+s(r)+1:64*chanum*m(r)+s(r))=g(r,1+s(r):32*chanum*m(r)+s(r));
g(r,64*chanum*m(r)+s(r)+1:128*chanum*m(r)+s(r))=g(r,1+s(r):64*chanum*m(r)+s(r));
end

% Define a new matrix “ga” which  is the transpose of matrix “g”
ga=g';

% Define a new matrix “gb” which is extended one more column to show the number of
% rows
gb(:,[2:(chanum+1)])=ga(:,[1:chanum]);
[sgbr,sgbc]=size(gb);
gb(:,1)=(1:1:sgbr)';

% Define a new matrix “gc” that gives the row number through the number of search
gc=gb;
gc(nsearch+1:sgbr,:)=[];
[sgcr,sgcc]=size(gc);

% Find the fundamental period to check this result with the formula  result of it
 xper=gc(1,[2:(chanum+1)]);
   for jj=chanum:sgcr;
         if gc(jj,[2:(chanum+1)])==xper
           funper=gc((jj-(chanum-1)),1);
           break
        end
   end

% Find the first redundancies in matrix “gc”
```

```
k=1;
for ii=chanum-1:nsearch;
   xrec=gc(ii,[2:(chanum+1)]);
   for ij=1:floor((nsearch-ii)/(2*chanum))
       jj=ii+2*chanum*ij;
            if gc(jj,[2:(chanum+1)])==xrec
            redun=gc(jj,1);
            % Define a new matrix "h" to put the first redundancies
            h(k,1)=ii;
            h(k,2)=redun;
            k=k+1;
            break
        end
   end
end

% Define a new matrix " hsort" to sort the redundancy matrix "h"
hsort=h;
[yoy,ioi]=sort(hsort);

% Define a new matrix " hsorted" to put the actual row number of  "hsort"
hsorted=[yoy(ioi(:,2),1) yoy(:,2)];
hsorted;

% Define a new matrix "hreduced" to eliminate the rows of the matrix "hsorted" and that
% will not allow the first column to be monotone increasing
[ssr,ssc]=size(hsorted);
hreduced=hsorted;
a=h(ssr,1);
```

```
[rx cx]=size(hreduced);
for k=1:ssr
   for i=1:ssr
      if i<rx
        if hreduced(i,1)==a
          hreduced(i+1:ssr,:)=[];
           break
           elseif hreduced(i+1,1)<hreduced(i,1)
           hreduced(i+1,:)=[];
           break
           end
      end
   [rx cx]=size(hreduced);
   end
end
hreduced;

% Define a new matrix "H" that shows the point index of the distinct vectors without
% redundancy and their lengths
 [hsr,hsc]=size(hreduced);
H(1,1)=(chanum-1);
H(2:hsr+1,1)=hreduced(1:hsr,1)+1;
H(1:hsr,2)=hreduced(1:hsr,2)-1;
H(hsr+1,2)=nsearch;
H(1:hsr+1,3)=H(1:hsr+1,2)-H(1:hsr+1,1)+1;
[Hsr,Hsc]=size(H);

% Find the point dynamic range of the RSNS and put these values to the matrix "HH"
t=0;
```

```
for n=1:Hsr
   for p=H(n,1):H(n+1,1)
      HH(p,1)=p;
      HH(p,2)=H(n,2);
      HH(p,3)=H(n,3)-t;
      t=t+1;
      if t==(H(n+1,1)-H(n,1)+1)
        t=0;
      end
   end
   n=n+1;
   if n==Hsr
     break
   end
end
[HHsr,HHsc]=size(HH);
HH(HHsr,:)=[];

% Plot the point dynamic range for  the entered channel and shift values of RSNS
figure(1)
plot(HH(:,1),HH(:,3)),grid
xlabel('THE  SHIFTED  CHANNEL  VALUES'),ylabel('THE  DYNAMIC  RANGE  OF
THE CHANNEL VALUES')
title(['THE  DYNAMIC  RANGE  DISTRIBUTION  OF',num2str(chanum),'-CHANNEL
RSNS'])
orient tall
%print

% Find the dynamic range of the entered channel and shift values of RSNS
```

```
HHH=max(H);
dynrange=HHH(:,3);

% Display the dynamic range, fundamental period, entered channel and shift values and
% matrix "H"  which shows the %  beginning-ending point index of the distinct vectors
% without redundancy and their lengths
disp(' ')
disp(['THE FUNDEMANTAL PERIOD IS ',num2str(funper),' '])
disp(' ')
disp(['THE DYNAMIC RANGE IS ',num2str(dynrange),' '])
fprintf('\n    BEGIN POINT INDEX  END POINT INDEX  DYNAMIC RANGE\n')
fprintf('%11.0f %16.0f %12.0f \n',H')
fprintf('\n    CHANNEL VALUES     SHIFT VALUES\n')
fprintf('%11.0f %17.0f \n',mm')
prompt=input('Would you like to try another shift (y/n) ? >>','s');
end
```

## 2. Digital Signal Processing for the RSNS ADC (spice.m)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% spice.m                                                                  %
% Modified July 10th 1997                                                  %
% This program finds the folded waveform and comparator outputs of         %
%  the 3-channel RSNSwhich is simulated in SPICE program                   %
%  m1 = 3      s1 = 0                                                      %
%  m2 = 4      s2 = 1                                                      %
%  m3 = 5      s3 = 2                                                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
% Load the folding circuit and comparator outputs of the modulus-3 RSNS
load newmod3.dat
```

```
a=newmod3;

% Plot the folding waveform and comparator outputs for the modulus-3 RSNS
figure(1)
subplot(4,1,1),plot(a(:,1),a(:,2),'y'),grid,title('MOD3'),ylabel('FOLDED OUTPUT')
subplot(4,1,2),plot(a(:,1),a(:,5),'r'),grid,ylabel('C3')
subplot(4,1,3),plot(a(:,1),a(:,4),'g'),grid,ylabel('C2')
subplot(4,1,4),plot(a(:,1),a(:,3),'b'),grid,ylabel('C1'),xlabel('ADC INPUT VOLTAGE')
orient tall

% Load the folding circuit and comparator outputs of the modulus-4 RSNS
load newmod4.dat
b=newmod4;

% Plot the folding waveform and comparator outputs for the modulus-4 RSNS
figure(2)
subplot(5,1,1),plot(b(:,1),b(:,2),'y'),grid,title('MOD4'),ylabel('FOLDED OUTPUT')
subplot(5,1,2),plot(b(:,1),b(:,6),'m'),grid,ylabel('C4')
subplot(5,1,3),plot(b(:,1),b(:,5),'r'),grid,ylabel('C3')
subplot(5,1,4),plot(b(:,1),b(:,4),'g'),grid,ylabel('C2')
subplot(5,1,5),plot(b(:,1),b(:,3),'b'),grid,ylabel('C1'),xlabel('ADC INPUT VOLTAGE')
orient tall

% Load the folding circuit and comparator outputs of the modulus-5 RSNS
load newmod5.dat
c=newmod5;

% Plot the folding waveform and comparator outputs for the modulus-5 RSNS
figure(3)
```

```
subplot(6,1,1),plot(c(:,1),c(:,2),'y'),grid,title('MOD5'),ylabel('FOLDED OUTPUT')
subplot(6,1,2),plot(c(:,1),c(:,7),'m'),grid,ylabel('C5')
subplot(6,1,3),plot(c(:,1),c(:,6),'r'),grid,ylabel('C4')
subplot(6,1,4),plot(c(:,1),c(:,5),'g'),grid,ylabel('C3')
subplot(6,1,5),plot(c(:,1),c(:,4),'c'),grid,ylabel('C2')
subplot(6,1,6),plot(c(:,1),c(:,3),'b'),grid,ylabel('C1'),xlabel('ADC INPUT VOLTAGE')
orient tall

% Plot the folding waveforms for modulus-3, 4, and 5 RSNS together
 figure(4)
subplot(3,1,1),plot(a(:,1),a(:,2),'y'),grid,title('MOD3'),ylabel('FOLDED OUTPUT')
subplot(3,1,2),plot(b(:,1),b(:,2),'y'),grid,title('MOD4'),ylabel('FOLDED OUTPUT')
subplot(3,1,3),plot(c(:,1),c(:,2),'y'),grid,title('MOD5'),ylabel('FOLDED OUTPUT')
orient tall
```

### 3. Transfer Function for the RSNS ADC (trans.m)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% trans.m                                                     %
% Modified July 10th 1997                                     %
% This program finds transfer function of the 3-channel RSNS, which is simulated %
% in   SPICE program.                                         %
% m1 = 3       s1 = 0                                         %
% m2 = 4       s2 = 1                                         %
% m3 = 5       s3 = 2                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
orient tall
% Initiate the variables to zero
i=0;
j=0;
```

```
l=0;
m=0;
n=0;
ka=1;
kb=1;
kc=1;
kd=1;
ke=1;

% Load the folding circuit and comparator outputs of the modulus-3 RSNS  and find
% howmany comparators are on
load newmod3.dat
a=newmod3;
[sar,sac]=size(a);
for i=1:sar;
   a1=a(i,3);
   a2=a(i,4);
   a3=a(i,5);
     if ( a1> 0);
        A(ka,1)=1;
     else
        A(ka,1)=0;
     end
     if ( a2 > 0);
        A(ka,2)=1;
     else
        A(ka,2)=0;
     end
     if ( a3 > 0);
```

```
        A(ka,3)=1;
      else
        A(ka,3)=0;
      end
      ka=ka+1;
   end
end

% Load the folding circuit and comparator outputs of the modulus-4 RSNS  and find
% howmany comparators are on
load newmod4.dat
b=newmod4;
[sbr,sbc]=size(b);
for j=1:sbr;
   b1=b(j,3);
   b2=b(j,4);
   b3=b(j,5);
   b4=b(j,6);
      if ( b1> 0);
        B(kb,1)=1;
      else
        B(kb,1)=0;
      end
      if ( b2 > 0);
        B(kb,2)=1;
      else
        B(kb,2)=0;
      end
      if ( b3> 0);
```

```
        B(kb,3)=1;
      else
        B(kb,3)=0;
      end
      if ( b4> 0);
        B(kb,4)=1;
      else
        B(kb,4)=0;
      end
      kb=kb+1;
   end
end

% Load the folding circuit and comparator outputs of the modulus-5 RSNS  and find
% howmany comparators are on
load newmod5.dat
c=newmod5;
[scr,scc]=size(c);
for l=1:scr;
   c1=c(l,3);
   c2=c(l,4);
   c3=c(l,5);
   c4=c(l,6);
   c5=c(l,7);
     if ( c1> 0);
        C(kc,1)=1;
     else
        C(kc,1)=0;
     end
```

```
      if ( c2 > 0);
        C(kc,2)=1;
      else
        C(kc,2)=0;
      end
      if ( c3> 0);
        C(kc,3)=1;
      else
        C(kc,3)=0;
      end
      if ( c4> 0);
        C(kc,4)=1;
      else
        C(kc,4)=0;
      end
      if ( c5> 0);
        C(kc,5)=1;
      else
        C(kc,5)=0;
      end
      kc=kc+1;
    end
end

% Define a new matrix "D"  that has every data of three folding circuit and shows how
% many comparators are on or off  in each of them
D(:,1)=a(:,1);
for m=1:sar;
    D(kd,2)=A(kd,1)+A(kd,2)+A(kd,3);
```

```
  D(kd,3)=B(kd,1)+B(kd,2)+B(kd,3)+B(kd,4);
  D(kd,4)=C(kd,1)+C(kd,2)+C(kd,3)+C(kd,4)+C(kd,5);
  kd=kd+1;
end

% Define a new matrix "DEC" that finds the transfer function
for ke=1:sar;
  aa=D(ke,2);
  bb=D(ke,3);
  cc=D(ke,4);
  dd=D(ke,1);
  if((aa==0)&(bb==0)&(cc==1))
    DEC(ke,1)=dd;
    DEC(ke,2)=-27;
  elseif((aa==0)&(bb==0)&(cc==0))
    DEC(ke,1)=dd;
    DEC(ke,2)=-26;
  elseif ((aa==1)&(bb==0)&(cc==0))
    DEC(ke,1)=dd;
    DEC(ke,2)=-25;
  elseif ((aa==1)&(bb==1)&(cc==0))
    DEC(ke,1)=dd;
    DEC(ke,2)=33;
  elseif ((aa==1)&(bb==1)&(cc==1))
    DEC(ke,1)=dd;
    DEC(ke,2)=34;
  elseif ((aa==2)&(bb==1)&(cc==1))
    DEC(ke,1)=dd;
    DEC(ke,2)=35;
```

```
elseif ((aa==2)&(bb==2)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=6;
elseif ((aa==2)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=7;
elseif ((aa==3)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=-24;
elseif ((aa==3)&(bb==3)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=-23;
elseif ((aa==3)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=22;
elseif ((aa==2)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=23;
elseif ((aa==2)&(bb==4)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=24;
elseif ((aa==2)&(bb==4)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-22;
elseif ((aa==1)&(bb==4)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-21;
elseif ((aa==1)&(bb==3)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-20;
elseif ((aa==1)&(bb==3)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=-19;
elseif ((aa==0)&(bb==3)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=-18;
elseif ((aa==0)&(bb==2)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=-17;
elseif ((aa==0)&(bb==2)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-16;
elseif ((aa==1)&(bb==2)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-15;
elseif ((aa==1)&(bb==1)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=15;
elseif ((aa==1)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=10;
elseif ((aa==2)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=41;
elseif ((aa==2)&(bb==0)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=-14;
elseif ((aa==2)&(bb==0)&(cc==2))
```

```
  DEC(ke,1)=dd;
  DEC(ke,2)=37;
elseif ((aa==3)&(bb==0)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=38;
elseif ((aa==3)&(bb==1)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=39;
elseif ((aa==3)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=-13;
elseif ((aa==2)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=35;
elseif ((aa==2)&(bb==2)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=6;
elseif ((aa==2)&(bb==2)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=-12;
elseif ((aa==1)&(bb==2)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=32;
elseif ((aa==1)&(bb==3)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=-11;
elseif ((aa==1)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=28;
```

```
elseif ((aa==0)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=29;
elseif ((aa==0)&(bb==4)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=-10;
elseif ((aa==0)&(bb==4)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=-9;
elseif ((aa==1)&(bb==4)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=26;
elseif ((aa==1)&(bb==3)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=27;
elseif ((aa==1)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=-8;
elseif ((aa==2)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=23;
elseif ((aa==2)&(bb==2)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=42;
elseif ((aa==2)&(bb==2)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=19;
elseif ((aa==3)&(bb==2)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=20;
elseif ((aa==3)&(bb==1)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-7;
elseif ((aa==3)&(bb==1)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=-6;
elseif ((aa==2)&(bb==1)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=17;
elseif ((aa==2)&(bb==0)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=-5;
elseif ((aa==2)&(bb==0)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=-4;
elseif ((aa==1)&(bb==0)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=14;
elseif ((aa==1)&(bb==1)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=15;
elseif ((aa==1)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=10;
elseif ((aa==0)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=11;
elseif ((aa==0)&(bb==2)&(cc==3))
```

```
  DEC(ke,1)=dd;
  DEC(ke,2)=-3;
elseif ((aa==0)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=-2;
elseif ((aa==1)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=8;
elseif ((aa==1)&(bb==3)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=27;
elseif ((aa==1)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=-1;
elseif ((aa==2)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=5;
elseif ((aa==2)&(bb==4)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=0;
elseif ((aa==2)&(bb==4)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=1;
elseif ((aa==3)&(bb==4)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=2;
elseif ((aa==3)&(bb==3)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=3;
```

```
elseif ((aa==3)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=4;
elseif ((aa==2)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=5;
elseif ((aa==2)&(bb==2)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=6;
elseif ((aa==2)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=7;
elseif ((aa==1)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=8;
elseif ((aa==1)&(bb==1)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=9;
elseif ((aa==1)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=10;
elseif ((aa==0)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=11;
elseif ((aa==0)&(bb==0)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=12;
elseif ((aa==0)&(bb==0)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=13;
elseif ((aa==1)&(bb==0)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=14;
elseif ((aa==1)&(bb==1)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=15;
elseif ((aa==1)&(bb==1)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=16;
elseif ((aa==2)&(bb==1)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=17;
elseif ((aa==2)&(bb==2)&(cc==5))
  DEC(ke,1)=dd;
  DEC(ke,2)=18;
elseif ((aa==2)&(bb==2)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=19;
elseif ((aa==3)&(bb==2)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=20;
elseif ((aa==3)&(bb==3)&(cc==4))
  DEC(ke,1)=dd;
  DEC(ke,2)=21;
elseif ((aa==3)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=22;
elseif ((aa==2)&(bb==3)&(cc==3))
```

```
  DEC(ke,1)=dd;
  DEC(ke,2)=23;
elseif ((aa==2)&(bb==4)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=24;
elseif ((aa==2)&(bb==4)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=25;
elseif ((aa==1)&(bb==4)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=26;
elseif ((aa==1)&(bb==3)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=27;
elseif ((aa==1)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=28;
elseif ((aa==0)&(bb==3)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=29;
elseif ((aa==0)&(bb==2)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=30;
elseif ((aa==0)&(bb==2)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=31;
elseif ((aa==1)&(bb==2)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=32;
```

```
elseif ((aa==1)&(bb==1)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=33;
elseif ((aa==1)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=34;
elseif ((aa==2)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=35;
elseif ((aa==2)&(bb==0)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=36;
elseif ((aa==2)&(bb==0)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=37;
elseif ((aa==3)&(bb==0)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=38;
elseif ((aa==3)&(bb==1)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=39;
elseif ((aa==3)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=40;
elseif ((aa==2)&(bb==1)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=41;
elseif ((aa==2)&(bb==2)&(cc==3))
  DEC(ke,1)=dd;
```

```
   DEC(ke,2)=42;
 elseif ((aa==2)&(bb==2)&(cc==4))
   DEC(ke,1)=dd;
   DEC(ke,2)=19;
 elseif ((aa==1)&(bb==2)&(cc==4))
   DEC(ke,1)=dd;
   DEC(ke,2)=-15;
 elseif ((aa==1)&(bb==3)&(cc==4))
   DEC(ke,1)=dd;
   DEC(ke,2)=-20;
 elseif ((aa==1)&(bb==3)&(cc==5))
   DEC(ke,1)=dd;
   DEC(ke,2)=-19;
 elseif ((aa==0)&(bb==3)&(cc==5))
   DEC(ke,1)=dd;
   DEC(ke,2)=-18;
 elseif ((aa==0)&(bb==4)&(cc==5))
   DEC(ke,1)=dd;
   DEC(ke,2)=43;
elseif ((aa==0)&(bb==4)&(cc==4))
   DEC(ke,1)=dd;
   DEC(ke,2)=44;
elseif ((aa==1)&(bb==4)&(cc==4))
   DEC(ke,1)=dd;
   DEC(ke,2)=-21;
elseif ((aa==1)&(bb==3)&(cc==4))
   DEC(ke,1)=dd;
   DEC(ke,2)=-20;
 elseif ((aa==1)&(bb==3)&(cc==3))
```

```
  DEC(ke,1)=dd;
  DEC(ke,2)=-8;
elseif ((aa==2)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=23;
elseif ((aa==2)&(bb==2)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=42;
elseif ((aa==2)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=7;
elseif ((aa==3)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=-24;
elseif ((aa==3)&(bb==1)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=39;
elseif ((aa==3)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=-13;
elseif ((aa==2)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=35;
elseif ((aa==2)&(bb==0)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=36;
elseif ((aa==2)&(bb==0)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=45;
```

```
elseif ((aa==1)&(bb==0)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=-25;
elseif ((aa==1)&(bb==1)&(cc==0))
  DEC(ke,1)=dd;
  DEC(ke,2)=33;
elseif ((aa==1)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=34;
elseif ((aa==0)&(bb==1)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=46;
elseif ((aa==0)&(bb==2)&(cc==1))
  DEC(ke,1)=dd;
  DEC(ke,2)=30;
elseif ((aa==0)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=-2;
elseif ((aa==1)&(bb==2)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=8;
elseif ((aa==1)&(bb==3)&(cc==2))
  DEC(ke,1)=dd;
  DEC(ke,2)=27;
elseif ((aa==1)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
  DEC(ke,2)=-8;
elseif ((aa==2)&(bb==3)&(cc==3))
  DEC(ke,1)=dd;
```

```
      DEC(ke,2)=23;
    end
    ke=ke+1;
    end
end

% Plot the transfer function of the 3-channel RSNS
figure(1)
plot(DEC(:,1),DEC(:,2)),grid
xlabel('ADC INPUT VOLTAGE'),ylabel('ADC DECIMAL OUTPUT CODE')
```

## LIST OF REFERENCES

[1] N.S. Szabo and R.J. Tanaka, *Residue Arithmetic and Its Applications to Computer Technology,* Mc Graw Hill Inc., New York, 1967

[2] I. Niven, H.S. Zuckerman and H.L. Montgomery, *An Introduction to The Theory of Numbers*, 5th Ed, John Wiley and Sons inc., New York, 1991

[3] G.E. Andrews, *Number Theory*, Dover Pub. Inc., New York, 1994

[4] H. Cohn, *Advanced Number Theory*, Dover Pub. Inc., New York, 1980

[5] K.H.Rosen, *Elementary Number Theory and Its Applications*, Addison and Wesley Pub. Inc., Canada, 1984

[6] P.E. Pace, R.E Leino and D. Styer, "Use of the Symmetrical Number System in Resolving Single-Frequency Undersampling Aliases," *IEEE Transactions on Signal Processing*, Vol. 45, no. 5, pp. 1153-1160, May 1997

[7] P.E. Pace, J.L. Schafer and D. Styer, "Optimum Analog Preprocessing for Folding ADCs," *IEEE Transaction on Circuits and Systems-II Analog to Digital Processing*, Vol. 42, no. 12, pp. 825-829, December 1995

[8] M.A. Soderstrand, W.K. Jenkins, G.A. Jullien and F.J. Taylor, *Residue Number System Arithmetic; Modern Applications in Digital Signal Processing,*IEEE Press, New York, 1986

[9] P.E. Pace, P.A. Ramamoorthy and D. Styer, "A Preprocessing Architecture for Resolution Enhancement in High-Speed Analog-to-Digital Converters," *IEEE Transactions on Circuits and Systems*, Vol. 41, no. 6, pp. 373-379, June 1994

[10] P.E. Pace and D. Styer , "High Resolution Encoding Process for An Integrated Optical Analog-to-Digital Converter," *IEEE Optical Engineering*, Vol. 33, no. 8, pp. 2638-2645, August 1994

[11] P.E. Pace, P.A. Ramamoorthy and D. Styer, "Resolution Enhancement for Guided-wave Analog-to-Digital Converters," *IEE Electronic Letters*, Vol. 28, pp. 2174-2175, June 1994

[12] R. Pieper, P.E. Pace, R. Van de Veire, and C. Foster, "Feasibility Demonstration of a High-Resolution Integrated Optical Analog-to-Digital Converter," *PSAA-III, 4th Annual ARPA Symp. On Photonic Systems for Antenna Applications,* Naval Postgraduate School, Monterey, CA, January 1994

[13] P.E. Pace and J. Schafer, "Decimation of Encoding Errors in an Optimum SNS Folding ADC," *IEEE Int'l Symp. On Circuit and Systems,* Seattle, WA, pp. 1324-1327, May 1995

[14] P.E. Pace and C. Foster, "High Resolution Direct Digitization of Multi-gigahertz Antenna Signals," *PSAA-III, 3rd Annual ARPA Symp. On Photonic Systems for Antenna Applications,* Naval Postgraduate School, Monterey, CA, January 1993

[15] P.E. Pace, R.D. Walley, R.J. Pieper, and J.P. Powers, "5-bit Guided-wave SNS Transfer Characteristics," *IEEE Letters*, Vol. 31, pp. 1799-1800, October 1995

[16] R.J. Van De Plassche and P. Baltus, "An 8-bit 100 MHz Full-Nyquist Analog-to-Digital Converter," *IEEE Journal of Solid State Circuits,* Vol. 23, no. 6, December 1988

## INITIAL DISTRIBUTION LIST

| | | No. Copies |
|---|---|---|
| 1. | Defense Technical Information Center<br>8725 John J. Kingman Rd., STE 0944<br>Ft. Belvoir, VA 22060-6218 | 2 |
| 2. | Dudley Knox Library<br>Naval Postgraduate School<br>411 Dyer Rd.<br>Monterey, CA 93943-5101 | 2 |
| 3. | Chairman, Code IW<br>Information Warfare Academic Group<br>Naval Postgraduate School<br>Monterey, CA 93943-5121 | 1 |
| 4. | Professor Phillip E. Pace<br>Department of Electrical and Computer Engineering<br>Naval Postgraduate School<br>Code EC/Pc<br>Monterey, CA 93943-5121 | 2 |
| 5. | Professor David Styer<br>Department of Mathematical Sciences<br>The University of Cincinnati, ML 25<br>Cincinnati, OH 45221 | 1 |
| 6. | Professor Jon Butler<br>Department of Electrical and Computer Engineering<br>Naval Postgraduate School<br>Code EC/Bu<br>Monterey, CA 93943-5121 | 1 |
| 7. | Professor David Jenn<br>Department of Electrical and Computer Engineering<br>Naval Postgraduate School<br>Code EC/Je<br>Monterey, CA 93943-5121 | 1 |

|  |  |  |
|---|---|---|
| 8. | Deniz Kuvvetleri Komutanligi<br>Personel Daire Baskanligi<br>Bakanliklar, Ankara, TURKEY | 2 |
| 9. | Deniz Harp Okulu Komutanligi<br>Kutuphane<br>Tuzla, Istanbul- 81704, TURKEY | 1 |
| 10. | Ilker Aydin AKIN<br>Cumhuriyet Mah. Igdeli Sok. No : 6<br>Bilecik, TURKEY | 2 |